ERROR MITIGATION IN NEXT GENERATION NAND FLASH MEMORIES

by

Mahmood Reza Alizadeh Ashrafi

B.S., Electrical and Telecommunications Engineering, University of Tabriz, 2009

M.S., Electrical and Electronics Engineering, Boğaziçi University, 2012

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Doctor of Philosophy

Graduate Program in Electrical and Electronics Engineering
Boğaziçi University
2018

ACKNOWLEDGEMENTS

First and foremost, I would like to sincerely thank my advisor Assoc. Prof. Ali Emre Pusane, who has patiently and with his immense knowledge led me through my entire research. I truly appreciate all his contributions of time, thoughts, and financial supports to make my Ph.D study an exciting and productive experience.

I appreciate Assoc. Prof. Şuayb Ş. Arslan for all his thoughtful and innovative comments, suggestions and discussions throughout my research.

I would also like to thank my other jury committee members, Prof. Hakan Deliç, Prof. Murat Saraçlar, and Assoc. Prof. Onur Kaya for their insightful comments, which have remarkably increased the quality of my Ph.D. dissertation.

I want to thank all my fellow labmates at the WCL, for creating a warm and friendly environment at our workplace.

I would like to show my deepest gratitude to my family: my beloved mother and father, and my lovely sisters Elnaz and Tannaz, and my smart nephew Souroosh for their eternal love, inspiration, and encouragement in my whole life. I also thank my father-in-law for his encouragement and support.

Last but not least, I thank my darling wife, Neda, who has been a constant source of support and encouragement during the challenges of graduate school and our life. I am truly grateful for having you in my life!

ABSTRACT

ERROR MITIGATION IN NEXT GENERATION NAND FLASH MEMORIES

The rapid developments and deployments in the global market of digital consumer electronics in the last decade have stimulated the universal research on the digital storage technologies. NAND flash memory, fabricated based on the floating gate transistor cells, has become the ubiquitous storage technology, dominating the market at tens of billion dollars annually, due to its advantageous properties including very fast operation, low cost, significant scalability potentials, and low power consumption. However due to its complex structure, flash memory suffers from several complicated non-stationary error sources, namely inter-cell interference, retention error, and random telegraph noise, which degrade its reliability and integrity. Recent advances in the next generation flash memories aiming to fabricate smaller and denser cells, specifically with introduction of multi-level cell technology capable of storing more than one bit of data per cell, result in amplified error sources, which critically limit the lifespan of the device. In this dissertation, error sources are analyzed and an accurate statistical model for the erroneously sensed threshold voltage of 2-level cells is derived, which enables us to compute the error rates of the flash memory channel in closed-form expressions. Furthermore, a novel two-dimensional equalization technique is proposed to mitigate the effect of the channel induced inter-cell interference. Derived expression are later extended to general cases, where the number of storage levels and interferers can arbitrarily be chosen. Finally, a novel machine learning based approach, based on both unsupervised and supervised learning mechanisms, is proposed, which can potentially tackle the non-stationarity of the channel to mitigate the corresponding error rates.

ÖZET

GELECEK NESİL NAND FLASH DEPOLAMA SİSTEMLERİNDE HATA AZALTMA

Son yıllarda tüketici elektroniğinin küresel pazarındaki hızlı gelişim ve dağıtımlar, dijital depolama teknolojileri üzerine yapılan araştırmaları canlandırdı. Kayan kapılı transistör hücrelerine dayanan NAND flash bellek, çok hızlı çalışma, düşük maliyet, önemli ölçeklenebilirlik potansiyelleri ve düşük güç tüketimi gibi avantajlı özellikleri sayesinde, her yıl onlarca milyar dolarlık pazarlara hitap eden, her yerde bulunan depolama teknolojisi haline gelmiştir. Bununla birlikte, karmaşık yapısından ötürü, flash bellek, aralarında hücre içi girişim, tutma hatası ve güvenilirlik ve bütünlüğünü bozan rastgele telgraf gürültüsü olmak üzere birkaç karmaşık hata kaynağından muzdariptir. Hücre başına birden fazla veriyi depolayabilen çok seviyeli hücre teknolojisinin tanıtılmasıyla daha küçük ve daha yoğun hücrelerin üretilmesini amaçlayan yeni nesil flash belleklerdeki son gelişmeler, bu cihazların kullanım ömrünü ciddi ölçüde sınırlandıran yükseltilmiş hata kaynaklarıyla sonuçlanır. Bu tez çalışmasında, hata kaynakları analiz edilmiş ve 2-seviyeli hücrelerin hatalı algılanan eşik voltajı için kesin bir istatistiksel model elde edilmiştir. Bu model flash bellek kanalının hata oranlarını kapalı biçimde ifade ederek hesaplamamızı sağlar. Ayrıca, kanalın yol açtığı hücre içi etkileşimin etkisini hafifletmek için yeni bir iki boyutlu dengeleme tekniği önerilmiştir. Türetilmiş ifade daha sonra genel durumlara genişletilmiş, depolama seviyesi ve müdahale edenlerin sayısı isteğe bağlı olarak seçilebilecek biçimde elde edilmiştir. Son olarak, denetimsiz ve denetimli öğrenme mekanizmalarına dayanan yeni bir makine öğrenimine dayalı yaklaşım önerilmiştir; bu yaklaşım, ilgili hata oranlarının azaltılması için kanalın durağanlaşmama olasılığını ortadan kaldırabilmektedir.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

$\mathbf{B}$	Base matrix
$\mathbf{H}$	Parity-check matrix
I	Number of pages
J	Number of strings
K	Number of interferers
$\mathcal{K}$	Kernel function of density estimation
L	Number of levels per cell
$\mathbf{m}$	Mean vector
n	Vector length
N_{PE}	Number of program/erase cycle
Q	Sensing resolution
s	Discrete symbol
S	Interference strength
T_r	Retention time
$\mathbf{v}$	Variance vector
V_e	Erased state voltage
V_P	Verification voltage
V_t	Threshold voltage
$\mathbf{w}$	Weight vector
x	Channel input term
$\mathbf{x}$	Training feature vector
y	Channel output term
y	Class label
z	Interference term

γ	Interference ratio
δ	Dirac delta function
ΔV	Incremental voltage step

λ	Parameter of the Laplace distribution
μ	Mean of the probability distribution function
ν	Additive white Gaussian noise
ν_r	Retention error
ν_t	Random telegraph noise
σ	Standard deviation of the probability distribution function
Φ	Mapping function of the support vector machine
ω	Bound of the truncated Gaussian distribution

LIST OF ACRONYMS/ABBREVIATIONS

1D	One Dimensional
2D	Two Dimensional
3D	Three Dimensional
AMISE	Asymptotic Mean Integrated Squared Error
BCH	Bose–Chaudhuri–Hocquenghem
CHE	Channel Hot Electron
CS	Computer Simulation
DNA	Deoxyribonucleic Acid
ECC	Error Correcting Code
EEPROM	Electrically Erasable Programmable Read Only Memory
EPROM	Erasable Programmable Read Only Memory
erf	Error Function
FG	Floating Gate
FN	Fowler-Nordheim
FTL	Flash Translation Layer
GA	Gaussian Approximation
HDD	Hard Disk Drive
ICI	Inter-Cell Interference
ISPP	Incremental Step Pulse Program
KDE	Kernel Density Estimation
LDPC	Low-Density Parity-Check
LLR	Log-Likelihood Ratio
MISE	Mean Integrated Squared Error
ML	Maximum Likelihood
MLC	Multi-Level Cell
MPC	Modified Post Compensation
nm	Nanometer
NVM	Non-Volatile Memory

OVO	One-Versus-One
PC	Post Compensation
PDF	Probability Distribution Function
PE	Program Erase
PROM	Programmable Read Only Memory
QP	Quadratic Programming
RAM	Random Access Memory
RBF	Radial Basis Function
ROM	Read Only Memory
RTN	Random Telegraph Noise
SC	Spatially Coupled
Seq. 1	Sequence Detection 1
Seq. 2	Sequence Detection 2
SER	Symbol Error Rate
SLC	Single Level Cell
SSD	Solid State Drive
SVM	Support Vector Machine
TI	Training Interval
TL	Training Length

1. INTRODUCTION

With the exponential increase in the amount of digital contents, which is mostly due to the ever-growing use of portable electronic devices, the demand for storage media is also increasing. For instance, it is estimated that by 2025, the humanity's accumulated digital data will increase to 163 Zettabytes (163 trillion gigabytes) [2]. Therefore, there has been a huge amount of industrial research interest in next-generation storage technologies to meet the overwhelming demand. Storage systems are generally divided into two categories: volatile and non-volatile memory (NVM). Stored data in volatile memory, e.g., random access memory (RAM), can be changed or refreshed, but it is lost when the power source is switched off. On the contrary, the most fundamental property of the NVM device, e.g. read only memory (ROM), is its capability to retain the stored data, even when the power source is shut off. Early versions of ROMs were fabricated with permanent data storage, i.e. it was not possible to later modify the content. Invention of programmable ROM (PROM) allowed users to program different contents via silicon/metal fuses. To program the data, corresponding fuses were blown, however this operation could only be performed once. In early 1970s, semiconductor-based erasable programmable ROM (EPROM) technology was introduced, which made the data erasing option possible via the ultraviolet light exposure. Since EPROM was based on solid-state electronics, compared to other available NVM technologies, it rendered high density and low-cost fabrication. However, the main breakthrough happened in early 1980, when the electrically erasable programmable ROM (EEPROM) was invented. The early generation of EEPROMs were bulky and slow, however they paved the way for the further developments, i.e. the introduction of faster and more scalable flash memories [3]. The name flash was given to this class of EEPROM, as the device is able to erase the whole chip data, similar to flash which lightens in a camera [4].

It is estimated that flash memories' global market will exceed \$60 billions in 2020 [5], which clearly shows the importance of research on the properties and reliability improvements of next generation flash memories. Currently, flash memories are getting smaller in dimensions faster than Moore's Law. However, due to the ultra-high density-scaling in the next-generation flash memory, several reliability issues have arisen, which may potentially jeopardize their future.

Deriving the statistical model of the flash memory's storage channel and necessary probability distributions, which basically generates essential prior information for detection and decoding and therefore directly affecting the reliability of the data storage, has been an important research subject in the relevant literature [1, 6–11]. However, a more accurate yet realistic model is still needed to be investigated, as previous studies fail to establish a proper approach, which simultaneously incorporates the full nature of various error sources. The goal of this dissertation is to establish an accurate statistical model of the flash memory's storage channel and to deal with serious challenges of the next-generation flash memory technology through demonstrating novel state-of-the-art approaches to overcome them.

1.1. Contributions of This Dissertation

This dissertation mainly focuses on the statistical modeling of the NAND flash memory's storage channel, to consequently propose novel signal processing based approaches for error mitigation in the storage channel. The main contributions of this work can be divided into three categories:

(i) An analytical approach to estimate the probability distribution of the flash memory's threshold voltage in an interference channel is carried out. These results are then extended to general cases, where the effect of all major error sources are incorporated and distributions and error probabilities are computed in closed-form expressions.

(ii) Novel efficient two-dimensional (2D) low-complexity equalization techniques are proposed to tackle the major interference problem.

(iii) Novel machine learning based approaches are proposed to tackle the non-stationarity of the flash memory's storage channel in a more intelligent manner.

In the remainder of this chapter, basics of the flash memory's structure are reviewed, in which the floating gate (FG) transistor and its operation principle are introduced. Different types of flash memories are described and their data storage mechanisms are discussed. Then, the reliability issues and major error sources in flash memory channel are reviewed in details. The focus point of the Chapter 2 is on the interference problem in flash memories, and to this end, first an analytical approach to estimate the probability distribution of the flash memory's threshold voltage is carried out, and later novel 2D equalization techniques, both hard detection and sequence detection based, are proposed and their performance efficiency is corroborated via extensive computer simulations. Furthermore, basics of error correcting coding (ECC) schemes for flash memories are reviewed and performance of new class of codes, i.e. spatially-coupled codes, are examined in the storage channel. Chapter 3 presents the extended results for the probability distribution of the sensed threshold voltage, where the effect of all major error sources are incorporated. In Chapter 4, novel machine learning based approaches are proposed to tackle the non-stationary nature of the flash memory's storage channel. The error sources of flash memories exhibit strong non-stationarity, which is dependent on several factors including number of usage cycles and retention time of the device. Operation temperature also affects the device's performance. Therefore, solely relying on the derived static statistical model is not adequate to fully analyze the system's behavior throughout its lifespan. Thus, by employing some basic techniques of machine learning, a periodic learning mechanism is proposed, which aims to mitigate the flash memory's erroneous effects in an intelligent way, and preliminary results corroborate the efficiency of the proposed methods. Finally, Chapter 5 concludes the dissertation.

1.2. NAND Flash Memory

A flash memory is a 2D structure consisting of many FG MOSFET transistors, which are similar to conventional MOSFET transistors, except there are two gate terminals instead of one, and the additional gate is called the floating gate [12]. Figure 1.1 shows the schematic of a FG transistor. By applying appropriate bias voltages,

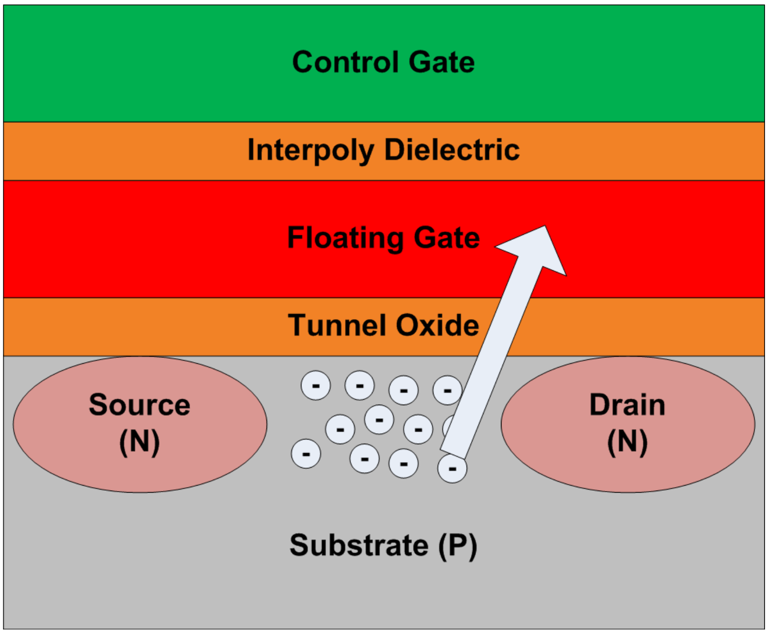

Figure 1.1. Schematic of the FG MODFET transistor.

electrons in the source-drain tunnel can pass through the oxide layer to reach the FG, and when the bias voltage is cut off, electrons are trapped in the FG. A similar procedure can be performed to sweep the trapped electrons in the FG back into the substrate tunnel. These actions causes the threshold voltage of the transistor (V_t) to change, and therefore different threshold voltage values can be quantized to represent different data.

Flash memories have been produced in two major fabrication architectures: NOR and NAND. NOR flash memory is relatively slow to write, however due to its fast random reads, it is mainly used for the BIOS data storage and code execution of operating systems. On the other hand, NAND flash memory, similar to the logical NAND gate, has serially-structured cells, which due to the less contacts per cell, results in improved density and therefore lower fabrication cost. However, their sequential structure results in an increased read latency. NAND flash memories are convenient for miscellaneous digital data storage, e.g. portable music players or cameras. However, their main utility is NAND flash based solid-state drives (SSD), which have recently become the viable storage media for portable computer, due to their high and very fast performance, low power consumption, low noise, and shock resistance capabilities,

Table 1.1. A comparison between different characteristics of NAND and NOR flash memories.

	NOR	NAND
Access	Random	Serial
Program latency	High	Low
Erase latency	High	High
Read latency	Low	High
Capacity	Low	High
Cost per bit	High	Low
Active power	High	Low
Standby power	Low	High
Endurance	Low	High
Cell size	Large	Small

which are conclusively replacing the obsolete hard disk drives (HDD).

The first commercial flash memory chip of NOR type was produced by Intel in 1988. The first NAND-type flash memory delivering a higher density, was manufactured by Toshiba in 1987. Initially, the growth of flash memory industry was relatively slow due to reliability and fabrication problems, however from early 2000 a considerable improvement has been continuously occurring in the manufacturing process and reliability aspects of flash technology. For instance, the fabrication technology in early 1990s was 700 nanometer (nm), which has currently scaled down to $1X$ nm ($10-19$ nm) [13,14]. Due to higher capacity of NAND flash memories and their less expensive fabrication, they has become the primary memory elements. Hence, this dissertation mainly focuses on the NAND flash memories. Table 1.1 summarizes the comparison between key characteristics of NOR and NAND flash memories [15]. Figure 1.2 shows the circuit level structure of the NAND flash memory. Each row and column are connected through wordline and bitline, respectively. Each row consists of one or more pages and each column is referred to as the string. A flash block is made of multiple pages, which is isolated from other block through a buffer page as shown in Figure 1.3.

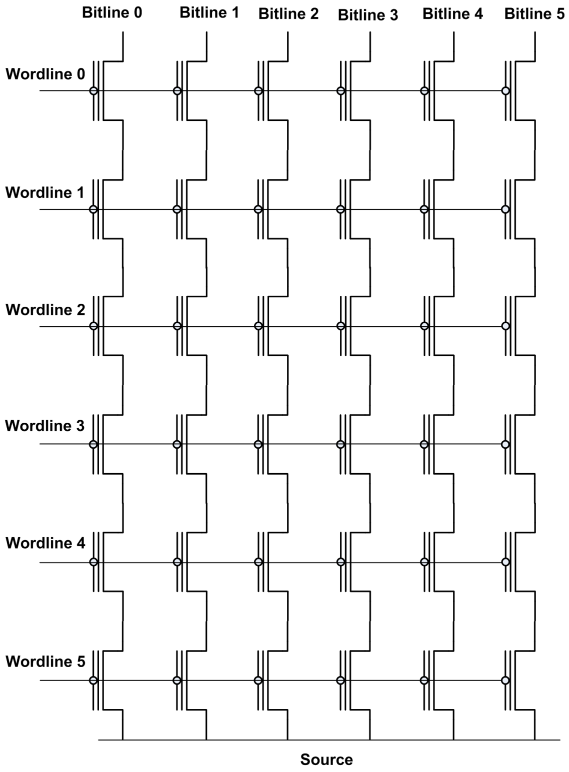

Figure 1.2. Circuit level schematic of the NAND flash memory block.

Each block usually has 64 to 128 wordlines and 2 KB to 8 KB bitlines. Each flash memory die consists of thousands of block.

Flash memory has three main functionality: erase, program (write), and read (sense). The programming basically controls the threshold voltage of the memory cell, which is achieved by the channel hot electrons (CHE) injection or the Fowler-Nordheim (FN) tunneling. The former approach is mainly used in NOR flash, while the latter one is employed in NAND flash [16]. In the CHE method, MOSFET is appropriately biased in drain and gate resulting in a large current flow through the cell, which energizes electrons in the channel to pass through the gate oxide barrier to the FG. FN tunneling is technically similar to the CHE injection, however only drain of transistor is biased, which produces smaller programming current. When the bias source shuts off, trapped electrons in the FG can no longer return to the channel, since they do not have enough energy to pass through the oxide barrier. Therefore, the presence (and amount) of electron charges in the FG can be employed as the data

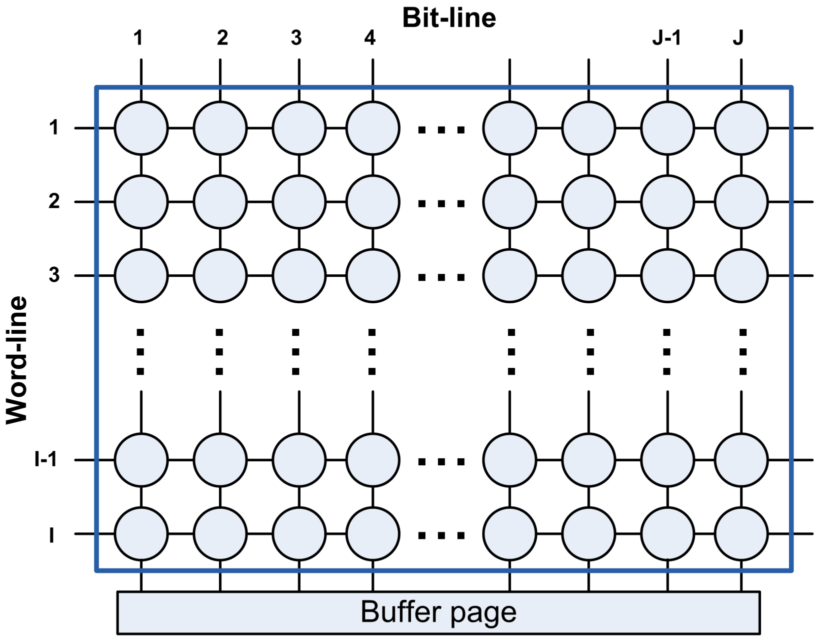

Figure 1.3. Hierarchy in the flash memory block.

indicator. The reverse operation can be applied to sweep electron off the FG back to the channel, which is referred to as the erasing the memory cell. Note that, FN tunneling is utilized for erasing in both NAND and NOR flash memories. Due to the different bias technique of the CHE and FN tunneling, power consumption and endurance of NAND and NOR flash memories are different, i.e. the CHE injection requires higher currents (in the order of milliAmpere), while the FN tunneling uses much less current per cell for programming (in the order of nanoAmpere). Also note that, since NAND flash memory uses FN tunneling for both erasing and programming, which is less likely to quickly degrade the MOSFET's oxide layer, its endurance is much better compared to the NOR flash memory [12].

The stored data can be read by sensing and measuring the threshold voltage of the FG transistor. To this end, current driven by the cell at a fixed gate bias is read. Different data values exhibit different threshold voltages V_t, which is proportional to the amount of electron charges trapped in the FG [12, 16].

Previous flash memories were based on the single level cell (SLC) technology, which as the name implies, can store one bit per memory cell. However, with the recent improvements of transistor scaling and reduced cost of manufacturing, the usage of multi-level cell (MLC) memory technology has grown quite dramatically, in which more than one bit of data can be stored via programming the threshold voltage V_t of a memory cell into one of the several non-overlapping voltage windows. In other words, data storage in an l-level-per-cell NAND flash memory, in which we can store $\log_2^l$ bits per cell, is achieved by programming the threshold voltage V_t of each memory cell into one of L non-overlapping voltage windows [12, 17].

The initial phase in a flash memory is the block erase operation, i.e., erasing the whole memory block before programming the memory cell. The erase operation is performed by setting the threshold voltage V_t to the lowest voltage level V_e, which is assumed to follow the Gaussian probability density function (PDF), i.e., $V_e \sim \mathcal{N}(\mu_e,\sigma_e^2)$ [1, 17]. In order to program a cell into a desired data value, its threshold voltage should be increased, which is achieved via the incremental step pulse program (ISPP), or program-and-verify technique [18], where the verification voltage of the target programming state is denoted by V_P. During each program-and-verify cycle, threshold voltage V_t is initially increased by a fixed voltage step ΔV, and then compared with the target voltage V_P. The procedure continues until V_t reaches V_P creating multiple flat voltage levels. This operation is illustrated in Figure 1.4. Due to the uniform nature of ISPP, programmed levels follow the uniform distribution over the interval $[V_{P_i} : V_{P_i} + \Delta V]$, for l-bit MLC where $i = 1, 2, \ldots, 2^l - 1$. Figure 1.5 shows the histogram of the written data in a 4-level ($l = 2$) flash memory, in which Gaussian distribution corresponding to the erased state and three different uniform intervals related to the rest of programmed states are depicted. As mentioned, programming is a page-oriented operation, therefore, erasing operation is much more costly and slower. Table 1.2 show the characteristics of three operations of flash memory, and as can seen, erasing is much slower [19].

Table 1.2. Characteristics of flash memory operations.

Operation	Unit	Latency
Erase	Block	2000 μSec
Program	Page	100-200 μSec
Read	Page	25-50 μSec

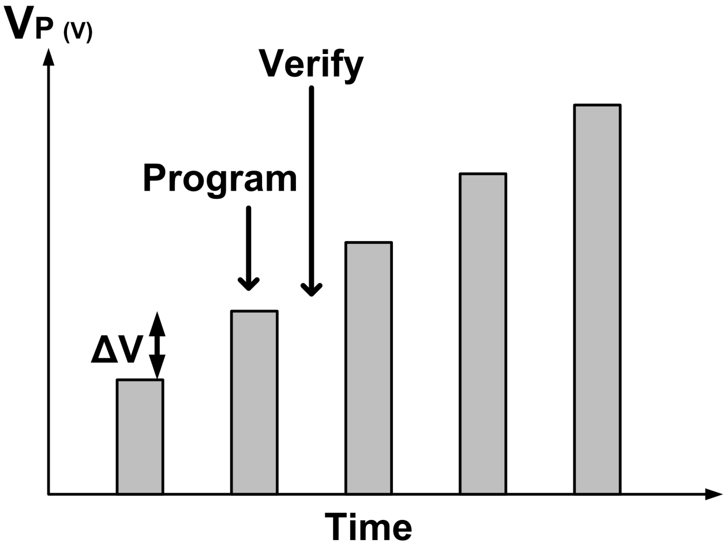

Figure 1.4. Memory cell programming by the ISPP.

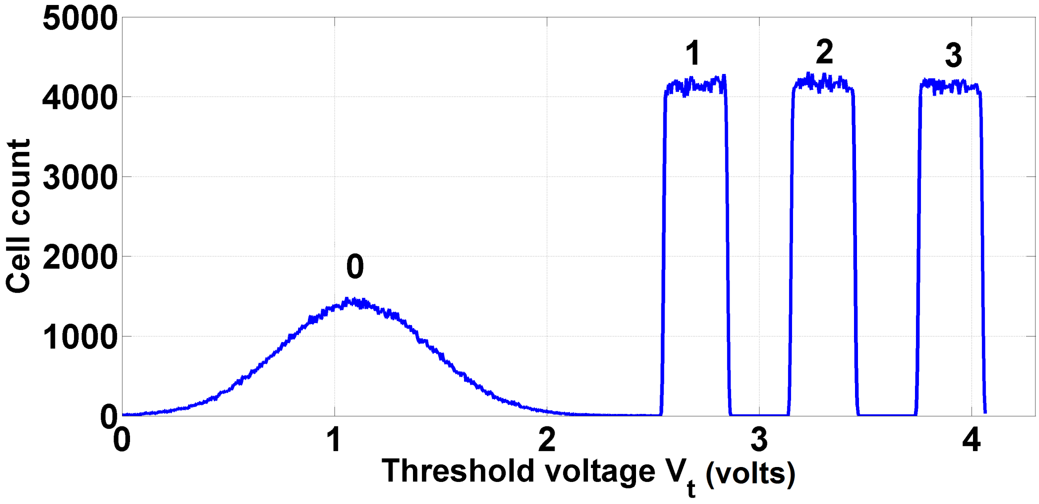

Figure 1.5. Simulated error-free cell-threshold voltage histogram for $l = 2$-bit MLC.

1.2.1. Programming Methods

As can be seen in Figure 1.2, each memory cell is surrounded by eight immediate neighbor cells. The threshold voltage of a specific cell can fluctuate based on its neighbors' stored data, which is due to the inter-cell interference (ICI) [20]. However, note that only few surrounding neighbors are actually considered as interferers, More clearly, only immediate neighbor cells, which are programmed after the victim cell, are assumed to interfere withe the victim cell. The next section covers the more detailed analysis of the ICI.

There are two architectures for programming flash memories. The first one is called even/odd bitline architecture (or shortly even/odd), in which cells on a page are divided into even and odd bitlines (strings). During the programming cycle, the cells at even bit-lines are written at the same time instant, and then the cells at odd bit-lines along the same word-line are programmed at the next time instant [17]. In this method, electronics components can be shared and reused, which brings several advantages including smaller, simpler, and cheaper fabrication circuitry. Considering the even/odd programming method, we realize that even-numbered cells and odd-numbered cells in a specific world-line have different number of interfering neighbors. This is illustrated in Figure 1.6. As can be seen from the figure, each even-numbered cell suffers from 5 interfering cells, while each odd-numbered cell has 3 interferers. Thus, they experience different levels of interference. The second emerging programming architecture is called the all-bitline method, in which all cells on a page are simultaneously written [9]. This improves the error performance of the flash memory, since each cell will suffer from three interferes. However on the other hand, it requires more complex and expensive circuitry. Note that in this dissertation, all-bitline programming has been assumed as the programming architecture of the flash memory.

Similar to conventional even/odd technique, another programming method called advanced even/odd is also proposed, in which, at the first page, cells at even bit-lines are written first and then, odd bitlines are programmed. Then, an entire page is skipped over and the third page is written. In other words, odd-numbered pages are

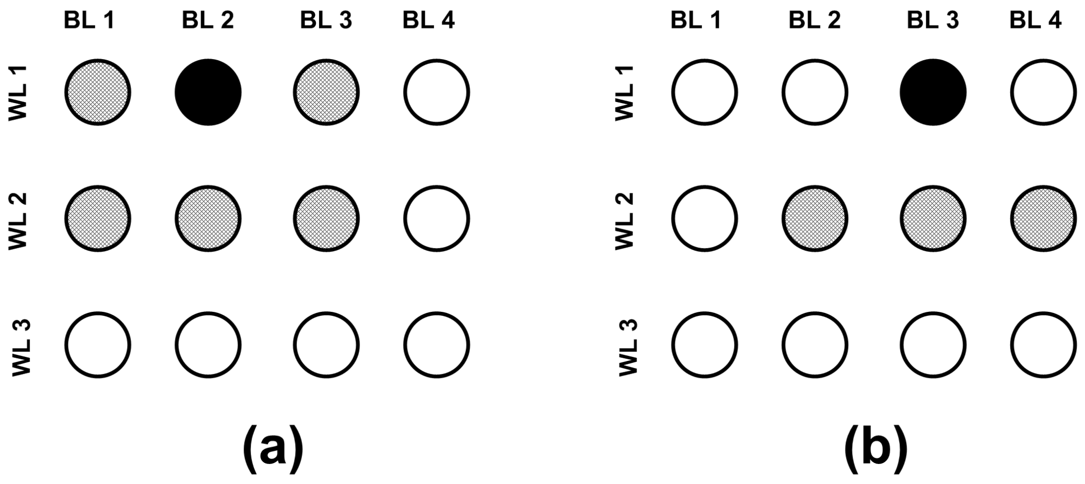

Figure 1.6. Schematic of the victim cell (black) and interferer cells (shaded) for (a) even cells and (b) odd cells in even/odd programming method.

programmed and when they are completely finished, the even-numbered pages are programmed. Thus, not only there is differentiation between even and odd bitlines, but also even and odd word-lines are also discriminated. In this algorithm, number of interferer cells for different positions within the block can be given as

$$N_{ij} = \begin{cases} 8 & i = 2k+1,\ j = 2k,\ k \in \mathbb{N}, \\ 6 & i = 2k+1,\ j = 2k+1,\ k \in \mathbb{N}, \\ 2 & i = 2k,\ j = 2k,\ k \in \mathbb{N}, \\ 0 & i = 2k,\ j = 2k+1,\ k \in \mathbb{N} \end{cases} \tag{1.1}$$

where N_{ij} shows the number of interfering cells for the victim cell located at the i^{th} page and j^{th} bit-line.

In the proposed advanced all-bit programming, all bits inside a page are programmed at once, which is similar to the conventional all-bit architecture. However, even-numbered pages are programmed only after all odd-numbered pages have already been programmed. A careful observation shows that in this method, cells in odd-numbered pages are affected by six interferer cells, while cell on an even-numbered page suffer from no ICI.

It is clear that the error rate of the all-bit programming outperforms the even/odd method, which is the result of programming order, in which, cells in the latter method experience higher levels of ICI on average. It also has been shown that the proposed advanced programming algorithms under severe ICI channel considerably outperform their conventional counterparts [21]. This is because of the fact that under a severe ICI environment, which is the likely case for the next-generation flash memories where the cell size considerably shrinks and thus ICI drastically increases, the data stored in all victim cells with interferer cells around, regardless of the number of interferers, is corrupted by intense ICI. However, in the proposed programming algorithms, some portion of the data can still be retained intact, since those cells experience no ICI. It is an important observation to note that the considerable improvements error performance is achieved with no extra computational complexity and circuitry burden, which make the proposed programming methods desirable to be implemented in the next-generation flash memories.

1.3. Error Sources

Flash memory cells deteriorate, while undergo continuous program/erase (PE) cycles. Repeated FN tunneling to write or erase data on/off the memory cell makes the oxide layer to accumulate charge holes and degrade. Therefore, trapped electrons in the FG can easily leak into the tunnel resulting in poorer data retention capability, more sensitivity to neighboring interference, and consequently prone to errors. As a result, all flash blocks experience a gradual and increasing error rate, as the PE cycle increases. Note that, vendors usually specify a specific block endurance for their products in terms of PE cycles, where the device is considered highly unreliable beyond this limit.

During the past decade, capacity of commercial flash memories has been increased thousands of times, which is due to the down-scaling fabrication and introduction of MLC technology. On the other hand, this enhancement results in the amplified circuit-level noises and errors, which severely threaten the reliability and endurance of the flash memory, and limits the lifespan of the device. With ongoing advances in the fabrication of smaller and denser memory cells, new challenges appears, which may decelerate the production process. For instance, modern flash memories are capable of storing roughly 100 electrons per cell [22], and quantizing the extremely small threshold value in several data windows needs a complex circuitry. On the other hand, losing only a couple of electron over the time may change the threshold voltage largely and degrade the error performance of the memory. Therefore, MLC memories have much shorter lifespan compared to their older SLC counterparts. For instance, the endurable PE cycles of MLC flash memory under the fabrication technology of $5X$-nm (50-59 nm) is about 10000, which has dropped to about 3000 for $2X$-nm technology [23]. This limited lifespan is the main concern for developing a fully flash memory based enterprise storage systems.

There are several error sources in the MLC flash channel memory, among which three issues are the dominant ones: inter-cell interference, retention error, and random-telegraph noise (RTN). The following subsections analyze these error sources and their statistical characteristics in details.

1.3.1. Inter-Cell Interference

ICI originates from parasitic capacitance coupling effects among the neighbor cells and results in the threshold voltage shift of the target cell while its interferers are being programmed [24, 25]. Interference ratio is defined as the ratio of the parasitic capacitance between the interferer and the victim cell, and the self-capacitance of the victim cell. As the fabrication technology continues to scale down, memory cells are packed closer to each other, and therefore the parasitic coupling capacitance between neighbor cells amplifies resulting in larger ICI. In fact, it has been shown that ICI is the major concern for the reliability of the next-generation NAND flash memories [26, 27]. Coupling takes place through the interference ratio γ_{ij} defined as $\gamma_{ij} = \frac{C_{ij}}{C_i}$, where C_{ij} is the parasitic capacitance between the interferer and the victim cell and C_i the capacitance of the victim cell itself.

Note that, the target cell is affected by interferers, which are programmed after the victim cell making the channel non-causal. It is assumed that immediate neighbors of a victim cell can affect the target cell, since the parasitic coupling quickly diminishes as the distance among cells grows [17].

As mentioned, in this dissertation all-bitline programming has been utilized, in which all cells on a page are programmed simultaneously, and each cell suffers from three neighbor aggressors located in the next page [28]. Hence, the total ICI comes from one vertical (V) and two diagonal (D) interferers. In general, the interfering effect of diagonal aggressors is assumed to be weaker compared to immediate vertical neighbor. Under this setup, the ICI of the victim cell is given as

$$z_{ij} = \sum_{k=-1}^{1} (x_{i+1,j+k} - V_{e_{i+1,j+k}})\gamma_{i+1,j+k} \tag{1.2}$$

It is observed that the ICI induces memory onto the flash storage channel. Note that, if the symbol stored in any of interfering cells is zero, there will be no interference toward the victim cell, as its voltage does not change from the erased level. Interference ratio, γ is assumed to comply with the truncated Gaussian distribution, due to the fact that only immediate interfering cells were considered as aggressor cells. Thus, $\gamma \sim \mathcal{TN}(\mu_\gamma, \sigma_\gamma^2, \omega)$, where ω determines the symmetric bound domain, i.e., PDF of γ is nonzero over the domain $\mu_\gamma \pm \omega$. Also, σ_γ and ω are assumed to be proportional to μ_γ [17].

It is also worth mentioning that characteristics of γ tend to moderately fluctuate due to PE cycles [25]. For the sake of comparison among variety of ICI ranges, latter values are scaled with ICI strength factor S. For instance, if $\mu_\gamma = \beta$, it is modified into $\mu_\gamma = \beta S$. Note that, higher values of S correspond to a more severe ICI, as intuitively expected.

1.3.2. Retention Error

The main functionality of the flash memory is to retain the stored data. However, due to the gradual and contiguous charge leakage into the tunnel, the threshold voltage gradually reduces, and hence stored data noticeably degrades over time [29]. As flash memory cell size shrinks with the next generation fabrication technologies, the number of trapped electrons decreases, and therefore the retention error begins to considerably dominate the read performance. Furthermore, this issue becomes more severe as the cell wears out under several PE cycles over time. The retention error voltage is typically modeled through a Gaussian PDF [30], that is

$$f_{\nu_{\mathbf{r}}}(\nu) = \frac{1}{\sqrt{2\pi\sigma_r^2}} \exp\left(-\frac{(\nu-\mu_r)^2}{2\sigma_r^2}\right) \tag{1.3}$$

Retention induced error voltage depends not only on PE cycles, but also on the programmed value, which makes it a data-dependent noise. Therefore, at a reference temperature, μ_r and σ_r can be modeled as follows [8]

$$\begin{aligned} \mu_r &= -C_r(V_P-\mu_e)N_{PE}^{\alpha}\Big(\ln{(1+\frac{T_r}{T_{r_0}})}\Big), \\ \sigma_r &= C_r'|\mu_r| \end{aligned} \tag{1.4}$$

where C_r, C_r', and α are technology-related constants depending on the material properties and fabrication process of the memory cell, V_P denotes the initial programmed voltage of the cell. N_{PE} shows the number of PE cycles. T_r is the retention time, and T_{r_0} is the unit time set to 1 hour.

Note that, a very strong correlation between retention time and the operation temperature has been reported, which can be formulated via Arrhenius law [29]. This law calculates the acceleration factor, i.e., the number of times that the retention time

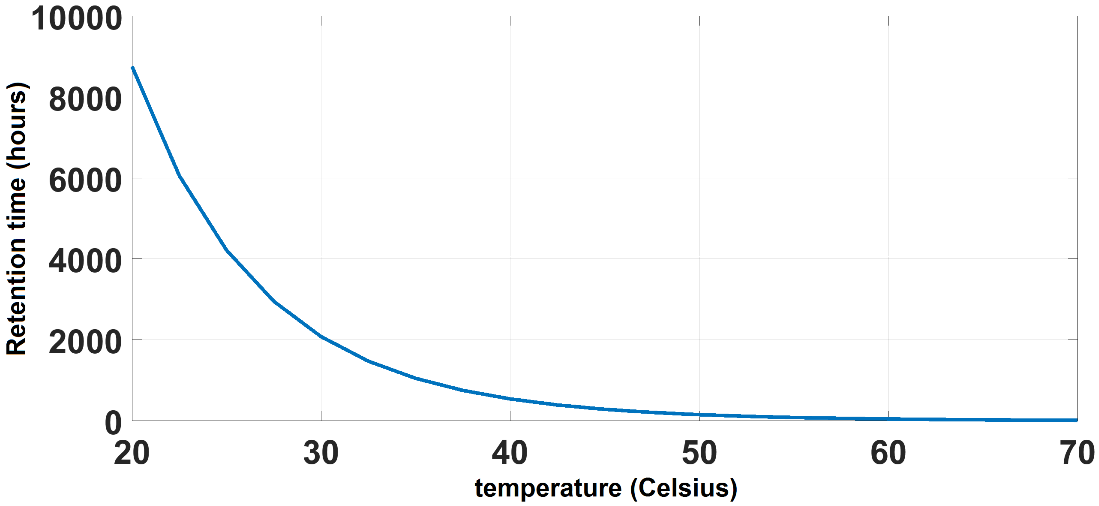

Figure 1.7. Retention time under various temperatures.

under the initial temperature decreases in the new temperature, which is given by

$$a_f = \exp\left(\frac{E_a}{K_B}\left(\frac{1}{t_{K_1}} - \frac{1}{t_{K_2}}\right)\right) \tag{1.5}$$

where E_a is the charge detrapping energy usually set to 1.1 eV for conventional flash memories, $K_B = 8.61733 \times 10^{-5}\ \frac{eV}{K}$ is the Boltzmann constant, and t_{K_1} and t_{K_2} show the initial and the final operation temperatures in Kelvin, respectively [31].

As an illustrative example, let us assume that the guaranteed retention time of a specific MLC device is one year (8760 hours) at 20^{oC} ($t_{K_1} = 293.15^{oK}$). Figure 1.7 shows the drastic decline of retention time of the same device under higher range of temperatures according to the Arrhenius model.

1.3.3. Random Telegraph Noise

RTN originates from random fluctuations in the MOSFET's drain current or threshold voltage, which is caused by electron capture and emission events at a charge trap site near the interface, and therefore is a major source of instability in flash memories [32]. The amplitude of RTN depends on fabrication and bias conditions, e.g. trap position and the channel area and the doping process of the substrate [33], and recent fabrication advances in the shrinkage of cell dimensions has amplified the RTN related degradation [34, 35]. Therefore, an accurate performance analysis of the next generation flash memories without integration of the RTN in the underlying model is not conceivable.

Based on a series of empirical experiments, it has been shown that the fluctuation magnitude follows the exponential decay. Therefore PDF of the threshold voltage fluctuation due to RTN is modeled as the symmetric exponential distribution, centered around zero Volt [36]. It is not hard to realize that this symmetric exponential distribution is indeed a zero-mean Laplace distribution with the PDF of

$$f_{\nu_t}(\nu) = \frac{1}{2\lambda} \exp\left(-\frac{|\nu|}{\lambda}\right) \tag{1.6}$$

where

$$\lambda = C_t N_{PE}^{\alpha} \tag{1.7}$$

and C_t is a technology-related constant. It is clear that the RTN exhibits a non-stationary behaviour through lifespan of the flash memory, as its intensity is correlated to the PE cycle of the memory.

1.4. Flash Translation Layer

In addition to solid state hardware implementations, flash memories usually include a software based mechanism, which is referred to as the flash translation layer (FTL). The main functionality of FTL is to establish a mapping between the physical storage system and the logical storage environment. In other words, FTL manages the logical block addresses of the stored data and the physical location of the data in flash chip. Therefore, FTL is capable of allocating the written data to an arbitrary location on a flash block. As discussed, memory blocks degrade over PE cycles, and therefore it is not desired to consecutively write data on a specific block while other blocks are constantly kept blank. Thus, FTL tries to spread programmed data out across the available flash space to improve the wear out leveling of different flash blocks.

As mentioned, flash memory based SSDs have become the primary storage devices. In addition to flash memory chips, they contain the controller chip, which is a high performance embedded processor unit. The SSD's FTL is located in the drive's controller, which includes software algorithms to create the synchronization interface between different flash memory chips and the disk requirement in the computer system. To this end, controller is responsible for efficient data storage to prolong the lifespan of the memory blocks via wear out leveling and bad block management, maximizing data transfer speed, and maintaining data integrity and device reliability through handling the ECC implementation [37].

2. TWO-DIMENSIONAL EQUALIZATION

Various studies have shown that among several error sources in multi-level cell memories, ICI is the most significant one. Therefore, to mitigate the devastating effect of the interference, simple, feasible, and yet efficient equalization techniques become essential for achieving desired data reliability. In this chapter, first a thorough analysis on deriving the distribution of the interference-free and interference-affected stored data is carried out. Then, novel low-complexity equalization methods are proposed, and their beneficial complexity-performance trade-offs compared to existing techniques are illustrated, where simulation results are presented to show that the proposed algorithms considerably improve the error performance, while maintaining the low-complexity constraints.

2.1. Error Probability Analysis

The channel model for the final sensed voltage of a cell is considered as

$$\begin{aligned} y_{ij} &= x_{ij} + \sum_{k=-1}^{1} (x_{i+1,j+k} - V_{e_{i+1,j+k}})\gamma_{i+1,j+k} + \nu_{ij} \\ &= x_{ij} + z_{ij} + \nu_{ij} \end{aligned} \tag{2.1}$$

where $x_{ij}, i = 1, \ldots, I, j = 1, \ldots, J$, denotes the target voltage of the cell located on i^{th} wordline and j^{th} bitline, representing equiprobable non-binary symbol $s_{ij} \in \{0, 1, 2, 3\}$, and γ_{ij} is the interference ratio. Note that, if the symbol stored in any of interfering cells is zero, there will be no interference toward the victim cell, as its voltage is not changed from the erased level. Currently for simplicity, the erroneous effects of all other deviating sources, e.g. retention noise and RTN, are lumped into the zero-mean additive white Gaussian noise samples ν_{ij}. Note that, the next chapter presents a more comprehensive error analysis. It is also worth mentioning that all of these parameters are random variables coming from different continuous distributions, which are assumed to be mutually independent. As previously indicated, the intended programmed cell voltage of input data has a mixture distribution consisting of four PDFs as given in

$$f_X(x) = \sum_{i=0}^{n-1} \Omega_i f_{X_i}(x) \tag{2.2}$$

where n and Ω_i denote the total number and weight of each individual PDF in the mixture model, respectively. Mean and variance of such a mixture model can be written as [38]

$$\begin{aligned} \mu_X &= \sum_{i=0}^{n-1} \Omega_i \mu_i, \\ \sigma_X^2 &= \sum_{i=0}^{n-1} \Omega_i \sigma_i^2 + \sum_{i=0}^{n-1} \Omega_i (\mu_i - \mu_X)^2 \end{aligned} \tag{2.3}$$

In the scenario of 4-level MLC flash memory, there are four PDFs present in the mixture, in which, $f_{X_0}(x)$ has the Gaussian distribution governing the erased state voltage (V_e), and the rest follows the uniform PDFs, which is illustrated in Figure 1.5. Due to equiprobability assumption on the input data sequence, mixture weights can be safely presumed to be $\Omega_i = \frac{1}{4}$.

As the initial step, we first analyze the ICI-free scenario, in which the sensed voltage simply becomes $y_{ij} = x_{ij} + \nu_{ij}$. It is well known that the PDF of the sum of independent random variables is the convolution of their marginal PDFs [39]. As a simpler case, summation of two independent Gaussian random variables from f_{X_0} and the zero-mean noise process ν results in the PDF $f_{Y_0}^{'} \sim \mathcal{N}(\mu_e, \sigma_e^2 + \sigma_\nu^2)$, where μ_e and σ_e^2 show the mean and variance of the erased state, respectively. The calculation of the rest of PDFs $f_{Y_i}^{'}$, $i \in \{1, 2, 3\}$ requires convolution of the Gaussian and uniform distributions, which results in

$$\begin{aligned} f_{Y_i}^{'}(y) &= \frac{1}{\Delta V} \left(\mathrm{Q}\left(\frac{y - (V_{p_i} + \Delta V)}{\sigma_\nu}\right) - \mathrm{Q}\left(\frac{y - V_{p_i}}{\sigma_\nu}\right) \right) \\ &= \frac{1}{2\Delta V} \left(\mathrm{erf}\left(\frac{y - V_{p_i}}{\sqrt{2\sigma_\nu^2}}\right) - \mathrm{erf}\left(\frac{y - (V_{p_i} + \Delta V)}{\sqrt{2\sigma_\nu^2}}\right) \right) \end{aligned} \tag{2.4}$$

Table 2.1. Simulation parameters.

Parameter	Value	Parameter	Value
V_P	[2.55, 3.15, 3.75] volts	μ_{γ_V}	0.08S
ΔV	0.3	σ_{γ_V}	$0.3\mu_{\gamma_V}$
μ_e	1.1	ω_{γ_V}	$0.2\mu_{\gamma_V}$
σ_e	0.35	μ_{γ_D}	0.006S
σ_ν	0.03	σ_{γ_D}	$0.3\mu_{\gamma_D}$
I , J	128, 2×10^4	ω_{γ_D}	$0.2\mu_{\gamma_D}$

where $\mathrm{Q}(\xi) = \frac{1}{\sqrt{2\pi}} \int_\xi^\infty \exp\left(\frac{-t^2}{2}\right) dt = \frac{1}{2} - \frac{1}{2}\,\mathrm{erf}\left(\frac{\xi}{\sqrt{2}}\right)$ and erf represents the error function.

Detection thresholds for reading are then simply calculated by finding the intersection points of these four PDFs, i.e., $\tau = [\tau_1 \ \tau_2 \ \tau_3]$. Therefore, the probability of symbol error can be derived as

$$\begin{aligned}
P(e) &= \sum_{i=0}^{3} P(e|f'_{Y_i})P(f'_{Y_i}) \\
&= \frac{1}{4}\Bigg(\int\limits_{y \notin (-\infty,\tau_1)} f'_{Y_0}(y)dy + \int\limits_{y \notin (\tau_1,\tau_2)} f'_{Y_1}(y)dy + \int\limits_{y \notin (\tau_2,\tau_3)} f'_{Y_2}(y)dy + \int\limits_{y \notin (\tau_3,\infty)} f'_{Y_3}(y)dy \Bigg) \\
&= \frac{1}{4}\Bigg(\int_{\tau_1}^{\infty} f'_{Y_0}(y)dy + \int_{-\infty}^{\tau_1} f'_{Y_1}(y)dy + \int_{\tau_2}^{\infty} f'_{Y_1}(t)dy \\
&+ \int_{-\infty}^{\tau_2} f'_{Y_2}(y)dy + \int_{\tau_3}^{\infty} f'_{Y_2}(y)dy + \int_{-\infty}^{\tau_3} f'_{Y_3}(y)dy \Bigg)
\end{aligned} \tag{2.5}$$

The calculation of the first integral is straightforward, thus, we focus on the integration of the erf function. Utilizing the technique of integration by parts, the integral is derived as

$$\int \mathrm{erf}\left(\frac{t-a}{b}\right) dt = (t-a)\,\mathrm{erf}\left(\frac{t-a}{b}\right) + b\frac{\exp\left(-\left(\frac{t-a}{b}\right)^2\right)}{\sqrt{\pi}} + C \tag{2.6}$$

where $C \in \mathbb{Z}$.

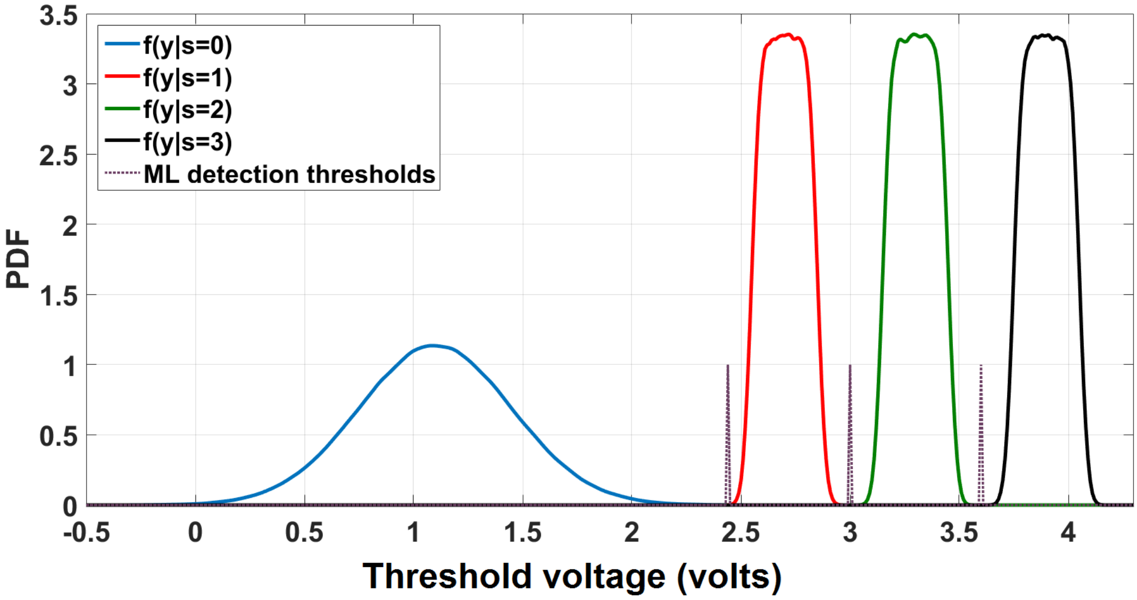

Figure 2.1. Conditional PDFs and ML detection thresholds for noisy but ICI-free channel ($S = 0$).

By employing the parameters listed in Table 2.1, the maximum likelihood (ML) optimal threshold values for ICI-free state are computed as $\tau = [2.44 \;\; 3.00 \;\; 3.60]$, which is depicted in Figure 2.1. Therefore error probability becomes $P(e) = 1.7561 \times 10^{-5}$, which represents the ICI free SER in the introduced flash channel under the employed parameters. Note that, in this section, the resolution of voltage sensing is limited to two decimal fraction digits.

The same analysis can be applied for the ICI channel. Note that, deriving the exact PDF of the channel output in Equation (2.1), in addition to summation, involves product of random variables, which can generally be computed via Mellin transform [40]. However, any attempt to compute the corresponding closed-form formula yields an analytically infeasible problem under the given setup, which originates from intractable integrals along with the presence of truncated Gaussian random variables. Thus, utilizing approximation tools to relieve the problem is inevitable. Here, Gaussian approximation (GA) is appropriately applied to the ICI term.

As discussed, the total ICI consists of three components, one vertical and two diagonal interferers, $z = z_V + z_{D_1} + z_{D_2}$, where z_{D_1} and z_{D_2} exhibit the same statistical characteristics. As mentioned earlier, if the interferer cell is programmed with zero, it does not interfere toward the victim cell, as its voltage does not change from the erased voltage floor. Thus, the effect of each interferer is either zero or a voltage shift, which we assume to follow the Gaussian distribution due to GA. Thus, it constructs a mixture model as

$$f_{Z_\bigstar}(z) = \frac{1}{4}\delta(z) + \frac{3}{4\sqrt{2\pi\sigma_\bigstar^2}} \exp(\frac{-(z-\mu_\bigstar)^2}{2\sigma_\bigstar^2}) \tag{2.7}$$

where $\bigstar \in \{V, D_1, D_2\}$, $\delta(z)$ is the Dirac delta function, and

$$\begin{aligned} \mu_\bigstar &= (\mu_{X'} - \mu_e)\mu_{\gamma_\bigstar}, \\ \sigma_\bigstar^2 &= (\sigma_{X'}^2 + \sigma_e^2)\sigma_{\gamma_\bigstar}^2 + (\sigma_{X'}^2 + \sigma_e^2)\mu_{\gamma_\bigstar}^2 + (\mu_{X'}^2 - \mu_e^2)\sigma_{\gamma_\bigstar}^2 \end{aligned} \tag{2.8}$$

Note that here, distribution $f_{X'}$ is a mixture of three uniform distributions as the erased state is previously excluded and separately handled. Therefore, parameters $\mu_{X'}$ and $\sigma_{X'}^2$ can be computed according to Equation (2.3).

$$\begin{aligned} \mu_{X'} &= \frac{1}{3}\sum_{i=1}^{3}\mu_{X_i'} = \frac{1}{3}(V_{P_1} + V_{P_2} + V_{P_3} + \frac{3\Delta V}{2}), \\ \sigma_{X'}^2 &= \frac{\Delta V^2}{12} + \frac{1}{3}\sum_{i=1}^{3}(\mu_{X_i'} - \mu_{X'})^2 \end{aligned} \tag{2.9}$$

Consequently, PDF of the total ICI can be written as

$$f_Z(z) = f_{Z_V}(z) * f_{Z_{D_1}}(z) * f_{Z_{D_2}}(z) \tag{2.10}$$

where $*$ denotes the convolution operator. Deriving the convolutions results in a mixture of Dirac delta function along with several Gaussian PDFs with different parameters

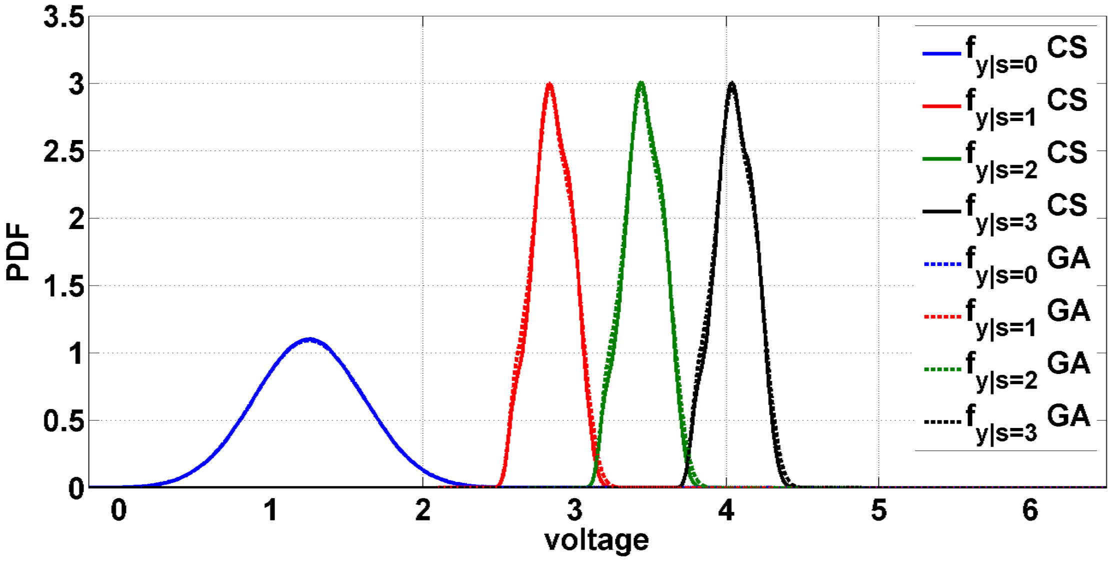

Figure 2.2. Conditional PDFs of the sensed voltage derived from CS and described GA process, where ICI strength factor is set to $S = 1$.

and different weights, that is,

$$f_Z(z) = w_0\delta(z) + \sum_j \frac{w_j}{\sqrt{2\pi\sigma_j^2}} \exp(-\frac{(z-\mu_j)^2}{2\sigma_j^2}) \tag{2.11}$$

where μ_j and σ_j^2 include sum of all possible combinations in sets $\{\mu_V, \mu_{D_1}, \mu_{D_2}\}$ and $\{\sigma_V^2, \sigma_{D_1}^2, \sigma_{D_2}^2\}$, respectively. The total number of sum of combinations in the set $\mathcal{S}$ is $2^{|\mathcal{S}|} - 1$, where $|\mathcal{S}|$ is the set's cardinality. Therefore, the total number of Gaussian terms in our expression is seven as the set contains three elements. Weight coefficients are calculated to be in the set $w_j \in \{\frac{1}{64}, \frac{3}{64}, \frac{9}{64}, \frac{27}{64}\}$, $j = 0, 1, \ldots, 7$. In other words, Equation (2.11) in a more appropriate form is given as

$$f_Z(z) = \sum_{j=1}^{8} \frac{3^{\lfloor \frac{j+1}{3} \rfloor}}{64\sqrt{2\pi v_j}} \exp\left(-\frac{(z-m_j)^2}{2v_j}\right) \tag{2.12}$$

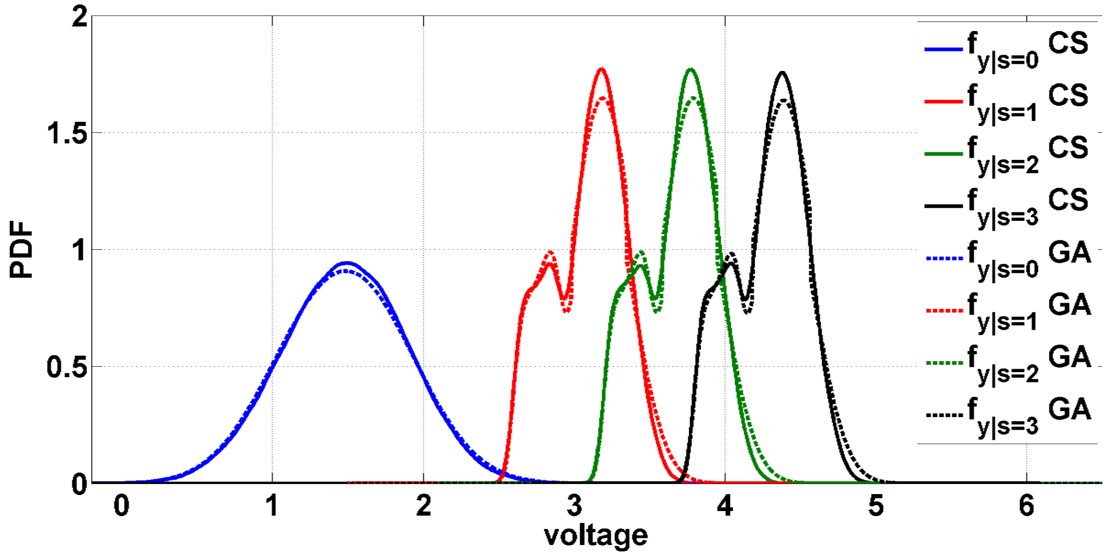

Figure 2.3. Conditional PDFs of the sensed voltage derived from CS and described GA process, where ICI strength factor is set to $S = 2.5$.

where $\lfloor \cdot \rfloor$ is the floor function, and

$$\mathbf{m} = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \times \begin{bmatrix} \mu_V \\ \mu_{D_1} \\ \mu_{D_2} \end{bmatrix} = \mathbf{P} \times \begin{bmatrix} \mu_V \\ \mu_{D_1} \\ \mu_{D_2} \end{bmatrix}, \tag{2.13}$$

$$\mathbf{v} = \mathbf{P} \times \begin{bmatrix} \sigma_V^2 \\ \sigma_{D_1}^2 \\ \sigma_{D_2}^2 \end{bmatrix}$$

Note that, for $j = 1$ in Equation (2.12), the mean and variance of the resulting Gaussian distribution equal to zero meaning that it tends to the Dirac delta function.

Now to derive the PDF of the final sensed voltages, considering the relation given in Equation (2.1), the resulted ICI PDF has to be convolved with the PDF of the noise and the input data voltage, that is,

$$f_Y(y) = f_Z(z) * f_X(x) * f_\nu(\nu) \tag{2.14}$$

Since the input data voltage can either follow the Gaussian or uniform distribution, we have to separately analyze these cases. Probability distribution of $f(y|s=0)$ is the convolution of two Gaussian PDFs with f_Z, which results in a mixture of several Gaussian PDFs with different weights and parameters, where the mean and the variance of each component is now the summation of the mean and the variance of the corresponding component of f_Z with those of the f_{Y_0} and f_ν.

$$f(y|s=0) = \sum_j \frac{w_j}{\sqrt{2\pi\sigma_j^2}} \exp(-\frac{(y-\mu_j)^2}{2\sigma_j^2}) \tag{2.15}$$

On the other hand, PDF of $f(y|s=i),\ i=\{1,2,3\}$ results in a mixture of erf functions described in Equation (2.4) with different weights and parameters.

To visualize the effect of approximation on the PDF of the sensed voltages, a comparison of four conditional PDFs is illustrated in Figure 2.2 and Figure 2.3, in which one set is derived from the flash channel computer simulation (CS), while the other is based on the Gaussian approximation procedure described above. As can be seen, the presented method renders an accurate and efficient estimation approach.

After deriving the PDF of the final sensed voltage, computing the optimal threshold values is straightforward, and therefore symbol error probability of the ICI channel can be determined in a closed-form rule according to Equation (2.5) and Equation (2.6), as each of conditional PDFs for the sensed voltage is either a weighted sum of the Gaussian or erf expressions with different parameters.

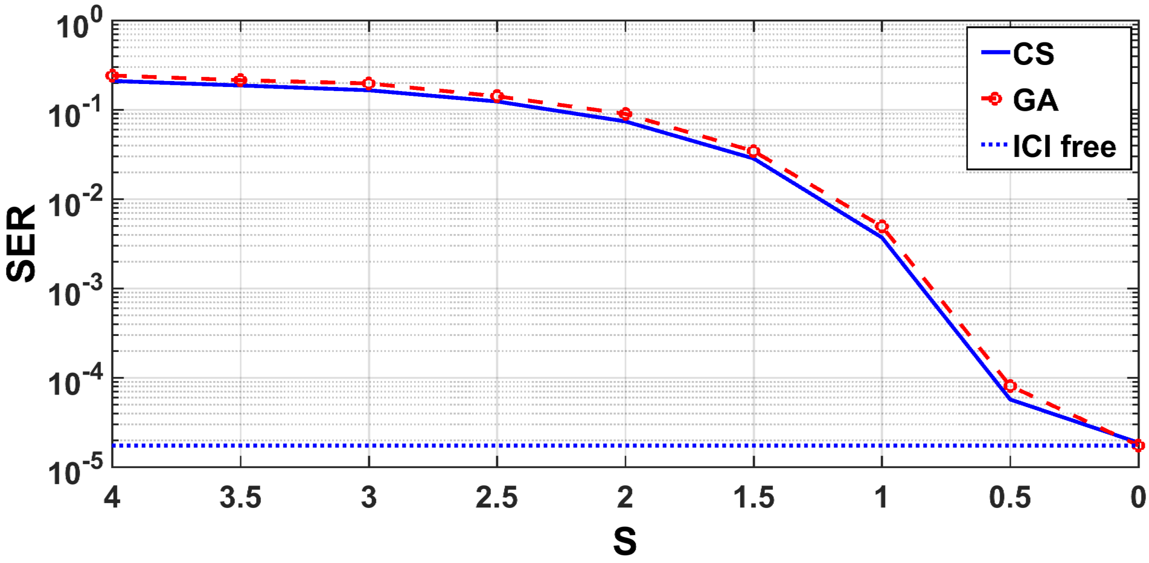

Figure 2.4. SER comparison under CS and GA.

Figure 2.4 demonstrates the comparison between SER values of the channel for various ICI strength factors (S), which are computed based on computer simulations (CS) and the afore-discussed approximation method, GA. SER of the ICI free channel is also presented.

2.2. Equalization

As discussed earlier, there are several error sources in the MLC memory channel, however, ICI has been determined to be the most critical one [20]. Due to the sophisticated 2D structure of memory arrays, classical equalization techniques, as the conventional approach to tackle the ICI, cannot be readily utilized. Therefore, deeper understanding of the flash channel and several modifications to available equalization techniques are inevitable. There have been several researches in the literature addressing the ICI problem in the flash memory channel to reduce the error rate. Authors in [41] and [28] have investigated the effect of different architectures of memory circuits and programming sequences on the improvement of error rates. Although these approaches indeed ameliorate the error performance of the flash memory, they have become insufficient as the research to design newer generations of flash memories with smaller yet denser cell sizes continues. Therefore, in addition to exploit the physical structure of the flash memories, more intelligent detection approaches have to be considered. Authors in [42], [43], and [44] have evaluated different methods to basically estimate the amount of threshold voltage shift caused by the ICI, and hence retrieve the original written data from the sensed voltage. However, their approaches lacks the realistic channel model, in which threshold voltage distribution under distinct voltage windows differs. In [17], a post-compensation (PC) method has been introduced and analyzed over a realistic flash channel. However, error performance of this approach deteriorates under severe ICI. All in all, to further improve the error performance, utilization of aforementioned hard-detection based approaches have to be surpassed. On this scope, there have been studies in the literature to investigate the sequence-detection algorithms over the 2D oriented channels [45,46]. However, the fundamental assumption lies on the separableness of the 2D channel, which basically means that the 2D channel response matrix can be differentiated into the product of two vectors, and hence allows the equalizer to handle the 2D ICI as the concatenation of two components: the one-dimensional (1D) row ICI and column ICI. Therefore, utilization of aforesaid method on the flash channel is not implementable, as the flash channel does not need to be separable.

Dealing with continuous random variables as the channel's input, along with the 2D structure of the memory block gives rise to a challenging problem of reliable data retrieving. As an optimal solution, a ML based equalization technique can be adopted in this scenario; however, designing the 2D ML equalizer is an NP-hard problem [47]. Therefore, simpler and feasible approaches should be investigated. As mentioned, in [17], the estimate of the ICI is computed and then subtracted from the sensed voltage to derive the estimation of the original written data voltage. The estimate of the ICI term is derived as

$$\widehat{z_{ij}} = \sum_{k=-1}^{1} (y_{i+1,j+k} - \mu_e)\mu_{\gamma_{i+1,j+k}} \tag{2.16}$$

It is seen that V_e and γ are estimated with their expected values. Therefore, the estimated programmed voltage can be written as $\widehat{x_{ij}} = y_{ij} - \widehat{z_{ij}}$.

The final phase after equalization is the demodulation, that is, mapping equalized voltages to discrete non-binary values based on demodulation thresholds. The first approach is to calculate optimal ML thresholds τ_{opt}, assuming access to training data to derive conditional PDFs via extensive computer simulations. These derived values are then assumed to be utilized through some embedded lookup tables. This path is convenient to demonstrate the theoretic potential of different equalization schemes; however, it does not seem practical enough in realistic implementations due to the probable lack of the training data. Therefore, a feasible approach, unaided demodulation, is also taken into consideration, in which threshold values are fixed at $\tau_u = [2.44 \;\; 3.00 \;\; 3.60]$ (tail intersection points of noisy but ICI-free PDFs discussed in the previous section).

Note that, based on the physical structure of flash memories, we can safely assume that the last page of a block will not suffer from interference, as there is no other page to be programmed afterwards (buffer page). This observation is important, because it enables an immediate demodulation of the original written data without any equalization process. This confident information can then be fed into the equalizer as the starting step. Thus, our proposed modified PC (MPC) equalizer begins the ICI estimation from the $I-1^{th}$ page, as the values of interfering cells from the last page can readily be utilized. Therefore, estimating x_{ij} from the sensed voltage y_{ij} is much more accurate than PC, and as each of lower pages feed their confident information to upper rows, a better error performance is achieved by MPC.

Figure 2.5 shows the SER of both aforementioned equalization techniques under optimal and unaided demodulations. ICI free SER is also demonstrated. It is seen that the MPC considerably improves the SER compared to the PC method, even though under the unaided demodulation, their error performance extremely declines. Therefore, to move beyond the hard-detection scope, the powerful low-complexity sequence-detection-based equalization method, as the main contribution of this section, is proposed.

The main emphasis is to initially reduce the channel into 1D judiciously, in order to address an analytically tractable problem. This is achieved by equalizing the effect

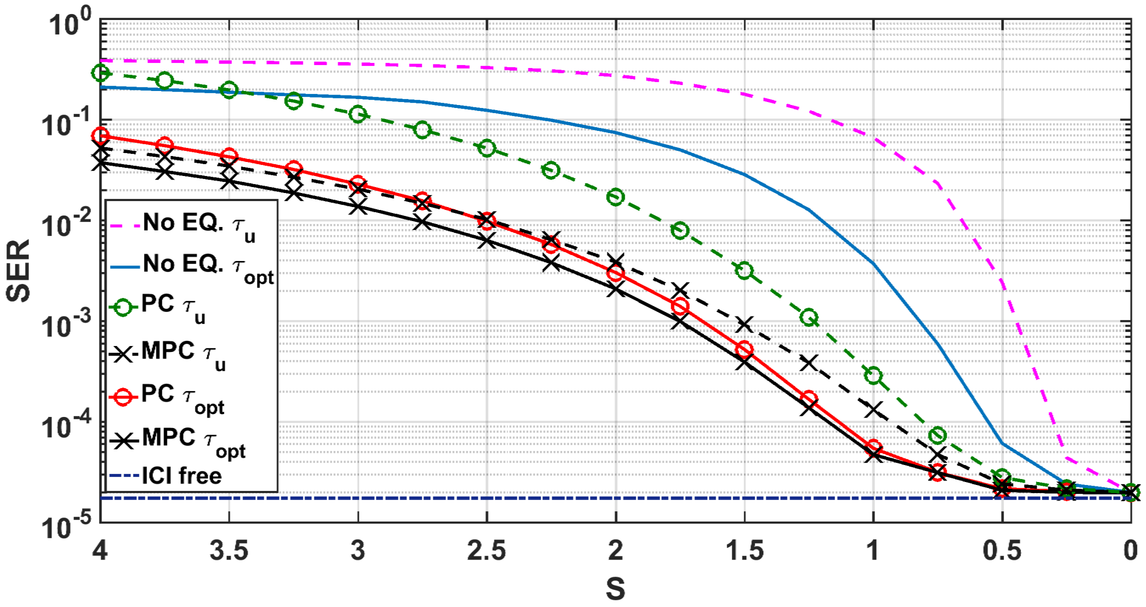

Figure 2.5. SER of the PC and the proposed method MPC.

of weaker diagonal aggressors through the PC method to produce a 1D signal [48]. Therefore, the resulting coarse-equalized 1D signal, assuming a perfect equalization of the diagonal ICI, is of form

$$y_{ij} = x_{ij} + (x_{i+1,j} - V_{e_{i+1,j}})\gamma_{i+1,j} + \nu_{ij} \tag{2.17}$$

Thus, column index can be simply neglected, that is, $y_i = x_i + (x_{i+1} - V_{e_{i+1}})\gamma_{i+1} + \nu_i$, which means that each bitline of a data block is individually processed. Note that, this equation describes a non-causal system. However, without loss of generality and for the sake of concordance, we can apply index flipping to build a causal system as

$$y_i = x_i + (x_{i-1} - V_{e_{i-1}})\gamma_{i-1} + \nu_i \tag{2.18}$$

For this 1D causal interference channel, the channel state according to Equation (2.18) is $\mathbf{S_T} = \{s_i s_{i-1}\}$. Therefore, state transition probability is $P(y_i|\mathbf{S_T}) = P(y_i|s_i s_{i-1})$. Now, let us assume the ML sequence-detection equalizer analogous to the Viterbi algorithm [49], which we here refer to as *Seq. 1*. The relation between the path metric and branch transition metrics between two states on the trellis in this equalizer is derived as

$$\begin{aligned} P(y_i y_{i-1} \dots y_1 | s_i s_{i-1} \dots s_1) &= P(y_1^i | s_1^i) \\ &= P(y_i | y_1^{i-1} s_1^i) P(y_1^{i-1} | s_1^i) \\ &= P(y_i | s_i s_{i-1}) P(y_1^{i-1} | s_1^{i-1}) \\ &= \prod_{i=1}^{I} P(y_i | s_i s_{i-1}) \end{aligned} \tag{2.19}$$

Note that, the third equality in Equation (2.19) comes from the fact that in our causal model, y_i is only dependent on s_i and s_{i-1}. For now, let us assume that probabilities $P(y_i|s_i s_{i-1})$ are derived (trained) via extensive computer simulations, and can be readily employed. Starting from the end of a column, which is assumed to be at the confident state $s_i = 1$, for each of four emerging paths all the way to the first page, metrics are calculated, and the survivor path carrying the maximum metric is selected. Note that, unaided cases, in which no prior information on the transition probabilities exists, is also considered for sequence-based equalizations. To this end, all 16 transition probabilities $P(y_i|s_i, s_{i-1})$ are computed via analytical Gaussian approximation method, in which different PDF cases can be categorized as follows.

$$\begin{aligned} \underset{j=0,k=\{1,2,3\}}{P(y_i|s_i = j, s_{i-1} = k)} &= \frac{1}{\sqrt{2\pi(\sigma_e^2 + \sigma_{z_k}^2 + \sigma_\nu^2)}} \exp\left(-\frac{\Big(y - (\mu_e + \mu_{z_k})\Big)^2}{2(\sigma_e^2 + \sigma_{z_k}^2 + \sigma_\nu^2)}\right), \\ \underset{j,k\in\{1,2,3\}}{P(y_i|s_i = j, s_{i-1} = k)} &= \frac{1}{2\Delta V}\left(\operatorname{erf}\left(\frac{y - (V_{p_j} + \mu_{z_k})}{\sqrt{2(\sigma_\nu^2 + \sigma_{z_k}^2)}}\right) - \operatorname{erf}\left(\frac{y - (V_{p_j} + \Delta V + \mu_{z_k})}{\sqrt{2(\sigma_\nu^2 + \sigma_{z_k}^2)}}\right)\right) \end{aligned} \tag{2.20}$$

Parameters μ_{z_k} and σ_{z_k} can be calculated as

$$\begin{aligned} \mu_{z_k} &= \left(V_{P_k} + \frac{\Delta V}{2} - \mu_e\right)\mu_{\gamma_V}, \\ \sigma_{z_k}^2 &= \left(\frac{\Delta V^2}{12} + \sigma_e^2\right)\sigma_{\gamma_V}^2 + \sigma_{\gamma_V}^2\left(V_{P_k} + \frac{\Delta V}{2} - \mu_e\right)^2 + \left(\frac{\Delta V^2}{12} + \sigma_e^2\right)\mu_{\gamma_V}^2 \end{aligned} \tag{2.21}$$

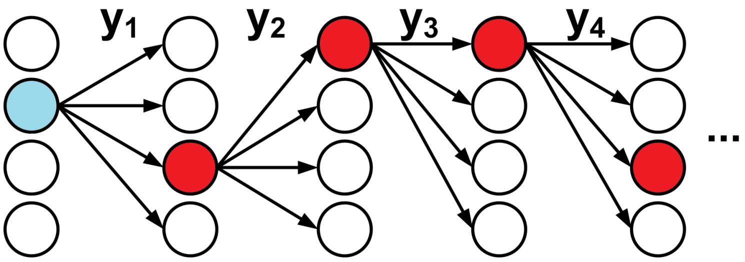

Figure 2.6. An illustration of the trellis based low-complexity equalizer. Starting state and decisive demodulated states are shown in blue and red, respectively.

As discussed earlier, $k = 0$ means that there is no interference, and therefore

$$\begin{aligned} P(y_i|\underset{j=0,k=0}{s_i = j, s_{i-1} = k}) &= \frac{1}{\sqrt{2\pi(\sigma_e^2+\sigma_\nu^2)}} \exp\left(-\frac{(y-\mu_e)^2}{2(\sigma_e^2+\sigma_\nu^2)}\right), \\ P(y_i|\underset{j\in\{1,2,3\},k=0}{s_i = j, s_{i-1} = k}) &= \frac{1}{2\Delta V}\left(\text{erf}\left(\frac{y-(V_{p_j})}{\sqrt{2\sigma_\nu^2}}\right) - \text{erf}\left(\frac{y-(V_{p_j}+\Delta V)}{\sqrt{2\sigma_\nu^2}}\right)\right) \end{aligned} \tag{2.22}$$

To reduce the equalizer's delay and complexity, another sequence-detection-based equalization scheme (*Seq. 2*) is proposed, in which the target symbol is demodulated at each time instant resulting in the noticeable reduced delay due to the fact that *Seq. 2* does not wait for the whole sequence to be processed for selecting the most likely sequence. Starting from the last row, whose corresponding symbol is readily demodulated to 1, the path having the maximum of four possible transition probabilities $P(y_i|s_i s_{i-1})$ is assumed to connect the currently-demodulated state (symbol) to the next one, which is then similarly demodulated, and the process continues as

$$\widehat{s_{i+1}} = \underset{s_{i+1}}{\text{argmax}}\, P(y_{i+1}|s_{i+1}\hat{s}_i) \tag{2.23}$$

This procedure is illustrated in Figure 2.6.

The comparison results of these two methods, under both prior knowledge assumption and GA (unaided), is given in Figure 2.7.

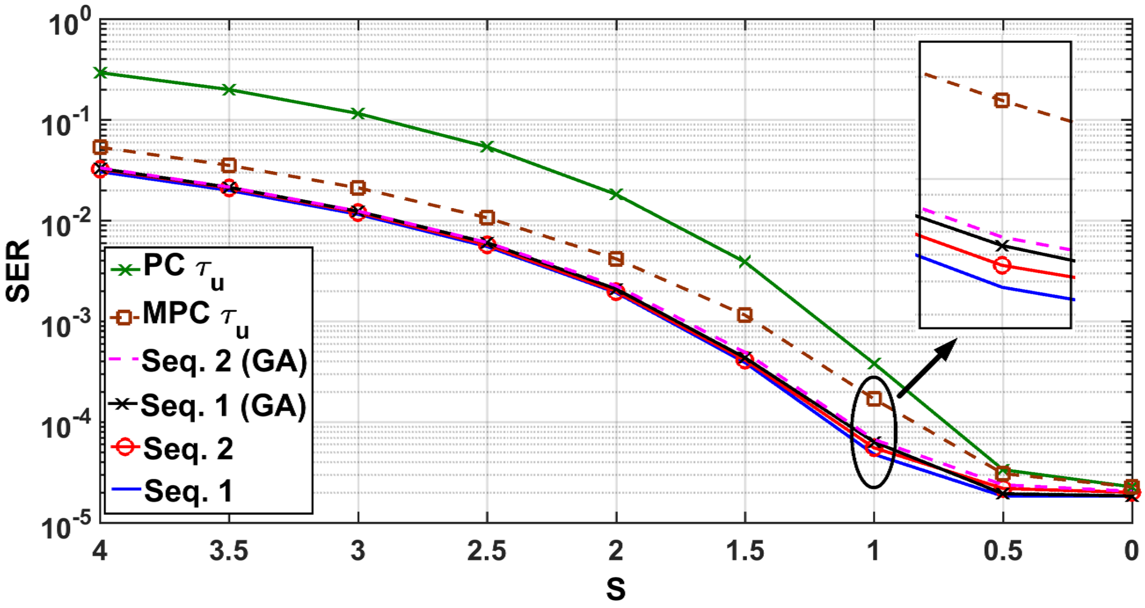

Figure 2.7. SER of the *Seq. 1* and the proposed method *Seq. 2*.

Carefully observing the curves, it is seen that *Seq. 1* negligibly outperforms *Seq. 2*; however, comparing computational complexity and the delay of the former method, employing the latter approach presents a very advantageous trade-off. Another important observation is the significant improvement of *Seq. 2*, specifically under the realistic unaided scenario, over PC and MPC techniques, which makes *Seq. 2* an effective low-complexity equalization technique for the flash channel.

To verify the eminence of the proposed equalization method, a very severe channel is generated based on the modification to some of simulation parameters in Table 2.1, i.e. $V_P = [2.55,\ 2.85,\ 3.15]$, $\Delta V = 0.15$, and $\gamma_{\mu_V} = 0.12S$. It is seen that the ISPP-based programming states have been narrowed and compressed close to each other, moreover the vertical interference has been enhanced to model a drastic channel environment. Figure 2.8 shows the SER results under this sever channel. As can be observed, *Seq. 2* still closely performers to the optimal equalizer *Seq. 1*, while noticeably outperforms the hard-detection based equalizers.

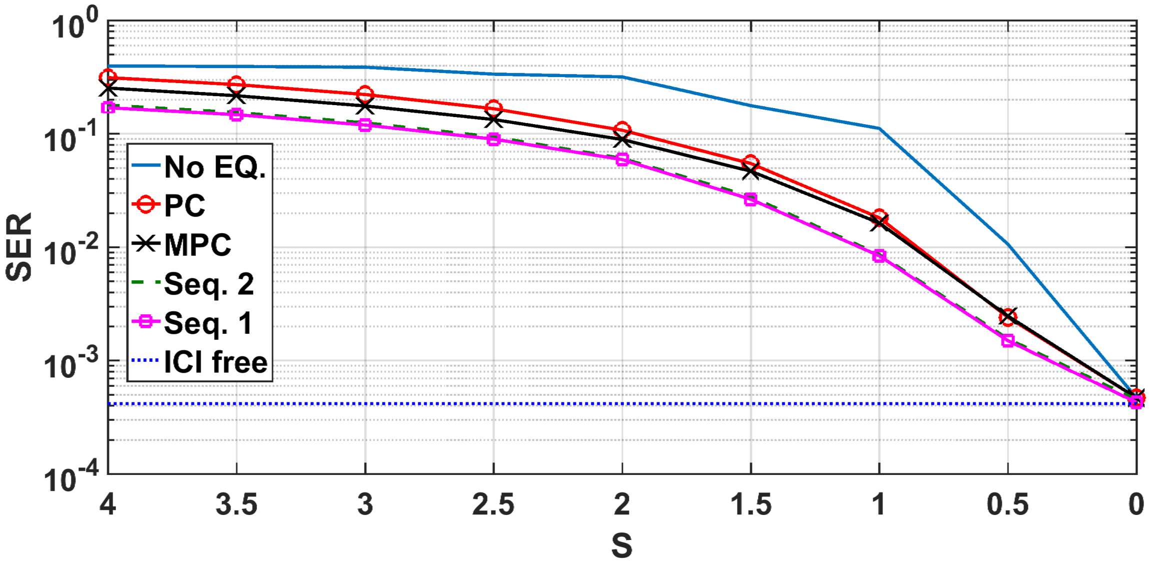

Figure 2.8. SER of aforementioned equalization schemes under the severe channel.

2.2.1. Delay Analysis

Page reading (access) latency is an important flash memory characteristic, which is typically $\eta \approx 25\,^{\mu Sec}$ for current MLC flash technologies [19]. Note that during the equalization process, not only the target page, but also other interfering pages has to be read, which results in more reading latency and overheads. For instance, the described PC method requires two consecutive page reads to equalize one specific page in the current setup, that is, page equalization latency is 2η. Note that, if the whole block is consecutively read, the resulting latency is equal to $I\eta$, as previously read pages can be reused for the equalization of upcoming pages.

For the proposed MPC and sequential-based equalizer however, the read latency for a single page is larger, as equalization is initiated from the last page and moves toward the first page. While the read delay is η for the last page, it becomes $I\eta$ for the first page. Therefore, the average read latency for a single page can be written as $(\frac{I+1}{2})\eta$. However, the accumulated reading latency for the whole block is still $I\eta$ for aforementioned methods. Considering the size of the current digital content to be written on flash memories, it is highly probable that the whole block (or at least a large portion of it) is consecutively programmed and read [17]. Therefore, under proposed equalization methods, a considerable error performance gain is achieved, while the read latency does not differ from that of the PC. Furthermore, if the content to be stored requires a single page programming/reading, to reduce the read latency, it can be written on the bottom pages, that is, the FTL can prioritize single-page contents to be written on lower pages.

As discussed, the hard-detection equalizers simply subtract the amount of estimated ICI induced voltage from the sensed voltage, that is, for each cell three estimated ICI voltages coming from two diagonal and one vertical interferes are subtracted from the output voltage, which results in a low-complexity equalization process. The proposed equalizer *Seq. 2* initially employs hard-detection method to equalize the weaker diagonal interferes to setup 1D equalization, and since the symbols are instantly demodulated according to Equation (2.23), it exhibits a low-delay characteristics compared to the viterbi-like sequence detection method *Seq. 1.* As discussed, transition probabilities can either be derived via extensive commuter simulations or simplified approximations, which then can readily be used from the stored lookup tables.

2.3. Error Correcting Codes for NAND Flash Memories

As a result of MLC technology and continuous shrinkage in the cell fabrication size, endurance and reliability of the next-generation flash memories degrades very quickly without consideration of additional techniques to compensate for these erroneous effects. As discussed in the previous section, equalization is an immediate signal processing tool to tackle the ICI as a major error source. However to achieve superior error performance, implementation of ECC techniques is inevitable. ECC schemes are capable of detecting and correcting errors at the cost of extra parity bits, and they currently have become an integrated segment of the flash memory's hardware architecture [50]. Early SLC NAND flash memories were equipped with simple single bit error-correction Hamming codes, to protect the data integrity, as SLC structure was robust enough against error sources [51]. For achieving a better error performance, flash vendors invested in Bose–Chaudhuri–Hocquenghem (BCH) codes, which render arbitrary level of error correction, along with being efficient in hardware implementation. This family of codes enjoys a relatively smaller parity data requirement, essential for production cost and processing latency, for achieving a certain error threshold, and they exhibit a guaranteed error-correction capacity t. For binary BCH codes of information and codeword lengths of (k, n), the following relations hold [52].

$$\begin{aligned} &n = 2^{\beta} - 1 \qquad \beta \in \mathbb{N}, \beta \geq 3 \\ &n - k \leq \beta t \\ &d_{\min} \geq 2t + 1 \end{aligned} \tag{2.24}$$

These ECC schemes are based on hard information and decision, which often limit their correction capability. Specifically, with introduction of MLC technology, capability of BCH codes has become seriously inadequate for data storage reliability. Therefore, it is highly motivated to implement more powerful ECC schemes, e.g., low-density parity-check (LDPC) codes, to further improve the reliability of flash memories [53,54]. Note that, unlike BCH codes, there is no known deterministic algorithm to compute the minimum distance of a particular LDPC codes due to their random structure, and therefore their error-correction capacity cannot be readily determined [55, 56]. The major issue in developing such sophisticated codes in flash memories is the soft information extraction. Although it has been shown that hard-information LDPC codes can outperform the corresponding BCH codes [57], their outstanding error performance directly relies on the soft information. To produce soft information, multiple sensing is required, which results in an increased read latency. Establishing a reasonable trade-off between the error performance of ECC schemes and read latency of flash memories is a major ongoing research in the industry.

2.3.1. LDPC Block Codes

Just like any other type of linear block codes, an LDPC block code of rate $R = k/n$ can be fully defined by the parity-check matrix $\mathbf{H}$ which has n columns and $m \geq n - k$ rows as there may be some linearly-dependent rows. However, this parity-check matrix $\mathbf{H}$ has low density. It means that in a binary LDPC code, number of 1s in the parity-check matrix $\mathbf{H}$ is very small compared to number of 0s. Another condition for sparsity of parity-check matrix **H** is that the number of 1s should be very small compared to m and n.

There are two types of LDPC block codes: regular and irregular codes. In regular codes, number of 1s is constant in each row and column of parity-check matrix **H** while in irregular codes, this number can vary for different rows and columns. An LDPC block code is said to be (J, K)-regular if there are exactly K 1s in each row and J 1s in each column of the parity-check matrix **H**. J and K are called column and row weight, respectively. For such codes, we can verify that $m \times K = n \times J$.

An LDPC block code can be illustrated by means of a graphical representation called the *Tanner* graph which sets a convenient basis for iterative decoding algorithm. A Tanner graph is a bipartite graph with two types of nodes: variable (bit) nodes and check nodes [58]. A variable node corresponds to the components of codewords and check node represents the set of parity-check equations.

There is a one-to-one correspondence between parity-check matrix **H** and the Tanner graph representation. If the variable node v_j $(j = 1, 2, \ldots, n)$ is connected to the check node c_i $(i = 1, 2, \ldots, m)$, then $H_{ij} = 1$, otherwise $H_{ij} = 0$. Sparse structure of LDPC codes enables an efficient iterative decoding to be performed on them. Iterative decoding is based on message-passing algorithm. As the name implies, certain messages are exchanged iteratively along the connecting edges between variable nodes and check nodes in the Tanner graph and at each round of iteration, messages are updated. These messages can be either hard-decision information such as binary values of 0 and 1 or soft information such as probabilistic values P_0 and P_1, which indicate probability of the certain bit being 0 and 1, respectively. Log-likelihood ratio (LLR) is also a suitable soft information which is defined as

$$LLR(v_i) = \log \frac{P(v_i = 0)}{P(v_i = 1)}, \quad i = 1, 2, \ldots, n \tag{2.25}$$

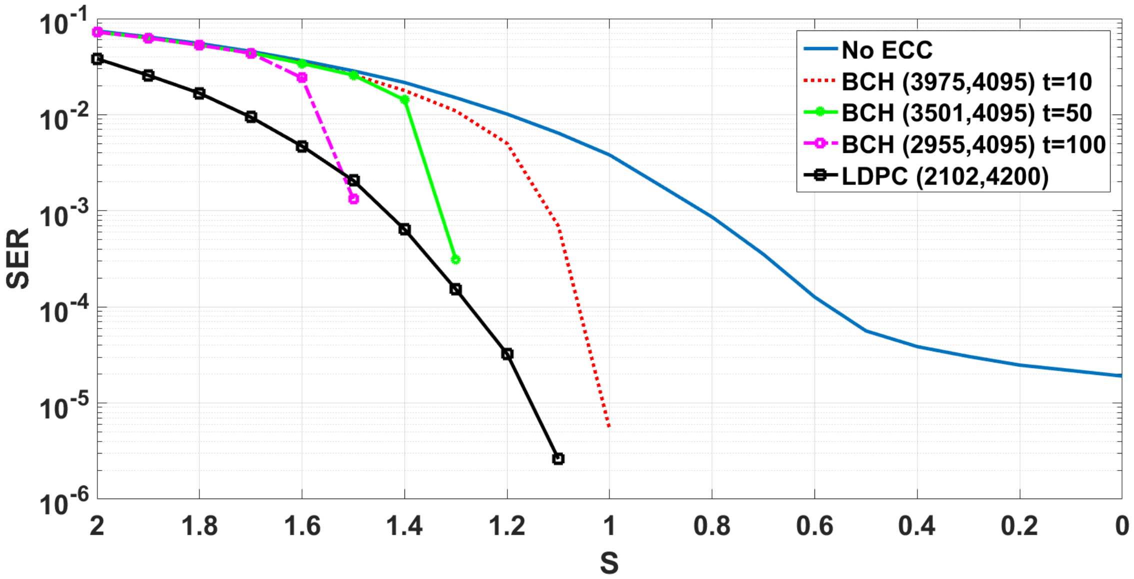

Figure 2.9. SER curves under different ECC schemes of (k, n) tuple and error correction capacity (t).

There are different algorithms based on message-passing including bit-flipping algorithm [59], belief-propagation (BP) [60], and min-sum which is the simplified version of BP [61]. Bit-flipping algorithm is based on hard-decision messages which results in less effective performance. Because of the trade-off between simplicity and performance of the algorithm, min-sum algorithm performs weaker than BP. For the sake of comparison, SER under three scenarios, no ECC scheme, BCH, and hard-information LDPC-coded data in the flash channel were carried out by computer simulation, which are shown in Figure 2.9. Note that for these set of simulation, non-binary data is first converted into binary sequence for encoding process, and then reconverted back to non-binary to pass the channel. Therefore, (K, N)-tuple represents the lengths of binary sequences. For the data reading, the reverse operation is applied. It is clear that ECC schemes considerably enhance the SER. Furthermore, we observe that by employing more powerful ECCs (larger t) error performance also improves. The employed LDPC code is a regular-$(3, 6)$ code, meaning parity of 50% which is a major drawback, specifically in the storage technology.

Topology of the Tanner graph is the key factor for message-passing-based decoding. A Tanner graph with a closed path (cycle) between some of variable nodes leads to failure of message-passing decoding. There several problematic concepts such as stopping set, trapping set, absorbing sets, and etc., which have to be taken into consideration to result a code with high performance.

The most common message-passing algorithm is the BP algorithm which is also known as the sum-product algorithm. The basic structure of the BP algorithm is the same as the bit-flipping algorithm, where messages are exchanged between variable and check nodes along the connecting edges of the Tanner graph representation iteratively. However, in BP, soft information (LLR values) are being exchanged. Similarly, there are two types of exchanged messages: a message from check node c_i to variable node v_j (q_{ij}) and the message from variable node v_j to check node c_i (r_{ji}). These messages can be calculated as

$$q_{ij} = \log \frac{1 + \prod_{j' \in N(i) \backslash j} \tanh(r_{j'i})}{1 - \prod_{j' \in N(i) \backslash j} \tanh(r_{j'i})}, \quad j \in N(i) \tag{2.26}$$

$$r_{ji} = LLR_j + \sum_{i' \in N(j) \backslash i} q_{i'j}, \quad i \in N(j) \tag{2.27}$$

In Equations above, $N(i)$ and $N(j)$ denote the neighbors of the node i and j, respectively. Symbol "$\backslash$" denotes the exclusion operation. For instance, $N(j) \backslash i$ means all neighbors of node j excluding node i. Initially, variable nodes send their received LLR values from the discrete channel to the connected check nodes, meaning that messages coming from check nodes are assumed to be zero. Then at each round of iteration, every check node collects messages from its connected variable nodes to calculate the response q_{ij} according to the Equation (2.26). Based on the received messages from different connected check nodes, each variable node calculates another message r_{ji} according to Equation (2.27) and sends it back to the interacted check nodes. This procedure takes place iteratively until the maximum number of iterations is reached. At this point, binary values of variable nodes (which indicate the bits of the codeword) are extracted by performing thresholding on the resulting LLR values.

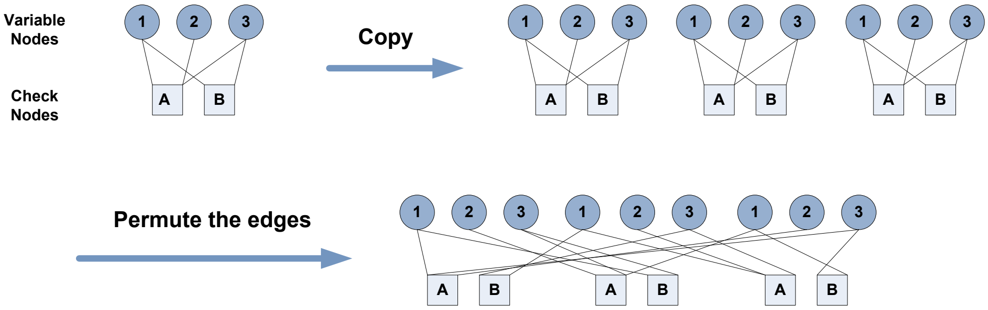

Figure 2.10. Copy-and-permute-operation.

2.3.2. Protograph Based SC Codes

The convolutional counterparts of LDPC block codes are called LDPC convolutional codes. LDPC convolutional codes are similarly defined by sparse parity-check matrices that also enable decoding using iterative message-passing algorithms [62]. Their main difference compared to LDPC block codes is the ribbon-shaped nonzero region in their parity-check matrices. This allows them to be used to construct a family of codes with variable frame lengths and low-complexity encoding/decoding properties. In this section, we consider a specific class of terminated LDPC convolutional codes, described by a parity-check matrix of the form

$$\mathbf{H}_{[L]} = \begin{bmatrix} \mathbf{H}_0(1) & & & \\ \mathbf{H}_1(1) & \mathbf{H}_0(2) & & \\ \vdots & \mathbf{H}_1(2) & \ddots & \\ \mathbf{H}_{m_s}(1) & \vdots & \ddots & \mathbf{H}_0(L) \\ & \mathbf{H}_{m_s}(2) & \ddots & \mathbf{H}_1(L) \\ & & & \vdots \\ & & & \mathbf{H}_{m_s}(L) \end{bmatrix} \tag{2.28}$$

The parity-check matrix structure given in Equation (2.28) is a representation of an LDPC convolutional code which has been terminated after L time instants and has a finite length. These codes are also called spatially-couple (SC) codes, due to the fact that they can be viewed as replacing a small LDPC block code and spatially coupling them. This introduces a memory among the small LDPC block codewords to be transmitted, which is called the code memory and is denoted by m_s. Since the elements of the matrix $\mathbf{H}_{[L]}$ are time dependent, the matrix will have time-varying properties.

This coding scheme has gained considerable research interest recently. Starting with introduction of LDPC convolutional codes in [63], there has been considerable attention paid to their high error performance. The research led to [64], where the authors reported higher iterative decoding threshold of these codes compared to those of the underlying LDPC block codes.

It was shown that SC code ensembles exhibit the so-called "threshold saturation" phenomenon, which means that their error performance, assuming BP decoding, can reach the ML decoding performance over the different communications channels [65], [66], and [67]. For this reason, a communication system that employs SC codes is expected to have high error performance, while exploiting several desirable characteristics.

A protograph is a small bipartite graph from which a larger graph can be obtained by a simple copy-and-permute procedure [68]. First, the protograph is copied M times, and then the edges of the individual copies are permuted to obtain a single large bipartite graph, referred to as the derived graph. The parameter M is called the expansion factor. If a protograph has N_P variable nodes and M_P check nodes, then the derived graph consists of $n = N_P \times M$ variable nodes and $m = M_P \times M$ check nodes. Figure 2.10 demonstrates this copy-and-permute operation for a simple example with $N_P = 3$, $M_P = 2$, and $M = 3$.

A protograph can be represented by an $M_P \times N_P$ biadjacency matrix, $\mathbf{B}$, which is called the base matrix of the protograph. Each entry in the base matrix, $\mathbf{B}_{i,j}$, $i = 1, \ldots, M_P$, $j = 1, \ldots, N_P$, is the number of edges between the corresponding check node c_i and variable node v_j. This is a non-negative integer and since parallel edges in protographs are allowed, it can be greater than one. SC codes can be constructed using protographs with base matrices in the form

$$\mathbf{B}_{[L]} = \overbrace{\begin{bmatrix} \mathbf{B}_0 & & & \\ \mathbf{B}_1 & \mathbf{B}_0 & & \\ \vdots & \mathbf{B}_1 & \ddots & \mathbf{B}_0 \\ \mathbf{B}_{m_\mathrm{s}} & \vdots & \ddots & \mathbf{B}_1 \\ & \mathbf{B}_{m_\mathrm{s}} & \ddots & \vdots \\ & & & \mathbf{B}_{m_\mathrm{s}} \end{bmatrix}}^{L} \tag{2.29}$$

where m_s is the code memory, each entry is a $J' \times K'$ submatrix, and for an asymptotically (J, K)-regular SC code family, the parameters J' and K' are obtained using $J' = J/\gcd(J, K)$ and $K' = K/\gcd(J, K)$. The corresponding protograph has $N_P = LK'$ variable nodes and $M_P = (L + m_s)J'$ check nodes, resulting in a code rate of

$$R_L = 1 - \frac{L + m_\mathrm{s}}{L}\frac{J'}{K'} = 1 - (1 + \frac{m_\mathrm{s}}{L})(1 - R) \tag{2.30}$$

where $R = 1 - (J'/K')$ is the rate of the unterminated LDPC convolutional code [69].

An example for the base matrix with $\mathbf{B}_0 = [2 \;\; 2]$ and $\mathbf{B}_1 = [1 \;\; 1]$, which is referred to as *ensemble B* [70], can be given as

$$\mathbf{B}_{[L]} = \begin{bmatrix} 2 & 2 & & & & & & & & \\ 1 & 1 & 2 & 2 & & & & & & \\ & & 1 & 1 & 2 & 2 & & & & \\ & & & & & \ddots & & & & \\ & & & & & & \ddots & & & \\ & & & & & & 1 & 1 & 2 & 2 \\ & & & & & & & & 1 & 1 \end{bmatrix} \tag{2.31}$$

Another example with $\mathbf{B}_0 = [2 \;\; 2]$, $\mathbf{B}_1 = [0 \;\; 1]$, and $\mathbf{B}_2 = [1 \;\; 0]$ which is called *ensemble D*, is given as

$$\mathbf{B}_{[L]} = \begin{bmatrix} 2 & 2 & & & & & & & & \\ 0 & 1 & 2 & 2 & & & & & & \\ 1 & 0 & 0 & 1 & 2 & 2 & & & & \\ & & 1 & 0 & 0 & 1 & 2 & 2 & & \\ & & & & & & \ddots & & & \\ & & & & & & & \ddots & & \\ & & & & 1 & 0 & 0 & 1 & 2 & 2 \\ & & & & & & 1 & 0 & 0 & 1 \\ & & & & & & & & 1 & 0 \end{bmatrix} \tag{2.32}$$

Note that while the code memory for the ensemble B is $m_s = 1$, $m_s = 2$ for ensemble D.

For a base matrix in this form, the copy-and-permute operation is equivalent to replacing each entry $\mathbf{B}_{i,j}$, $i = 1, \ldots, M_P$, $j = 1, \ldots, N_P$, in the base matrix with a sum of $\mathbf{B}_{i,j}$ permutation matrices of size $M \times M$. As a result, in (2.31) and (2.32), each of the entries 1 and 2 is replaced by an $M \times M$ matrix such that each column (and row) contains only 1 and 2 ones, respectively. Each 0 entry is replaced with an $M \times M$ zero matrix. The resulting matrix after this procedure represents the parity-check matrix $\mathbf{H}_{[L]}$ of the code, with size of

$$\mathbf{H}_{[L]} \sim (L + m_{\mathrm{s}})M \times 2LM \tag{2.33}$$

Constraint length of an SC code is also defined as

$$\nu_{SC} = (m_s + 1)K'M \tag{2.34}$$

In other words, constraint length ν_{SC} shows the row width of the non-zero region in the parity-check matrix $\mathbf{H}_{[L]}$ [71]. Note that decoder-complexity-wise, the performance of

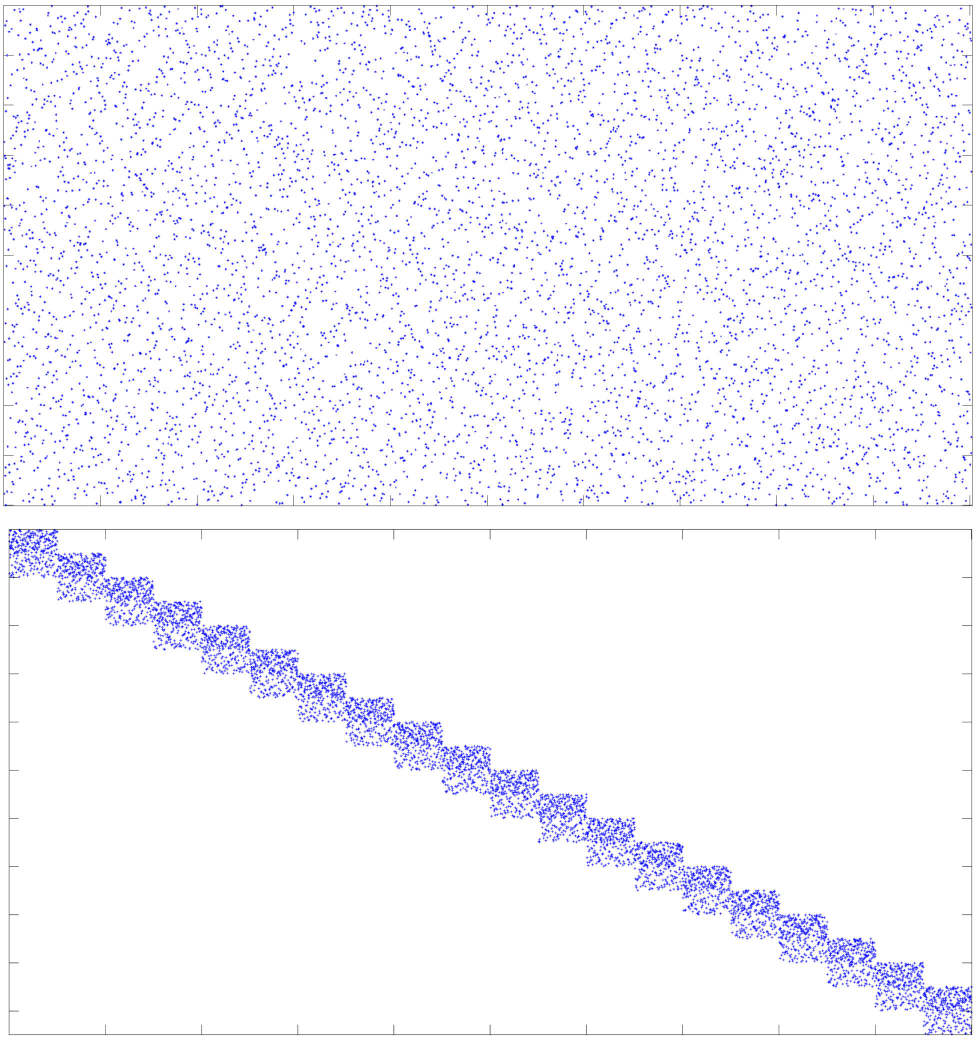

Figure 2.11. Sparsity visualization of the parity-check matrix of a regular block LDPC code (top), and regular SC code (bottom).

an LDPC block code is fairly comparable to its SC counterpart of constraint length ν_{SC} [72].

The ribbon-shaped structure of the parity-check matrix of the SC code, as illustrated in Figure 2.11, results in several observations. The first one relates to the degrees of the check nodes that are placed in the first and last m_{s} positions. Due to the special structure of the matrix, these check nodes have fewer connections to variable nodes, which, in turn results in a special irregularity of the row weights. This slight irregularity of SC codes resulting from the termination of the underlying LDPC convolutional code has been shown to be the main reason for SC codes having better BP decoding thresholds compared to corresponding block code ensembles [66].

This structure also can possibly allow SC codes to be decoded via a low-delay decoding scheme called window decoding [69]. In contrast to a block decoder, the window decoder does not need to wait for the entire frame to arrive at the receiver before starting the decoding process, which considerably lowers the decoding delay. In other words, it can start the decoding process once a small portion of the codeword is received at the decoder.

As mentioned, LLR is the key component for soft information based signal detection and decoding. For simpler channels, LLR values can easily be calculated in closed-form expressions. However, in our case, the problem is not that straightforward.

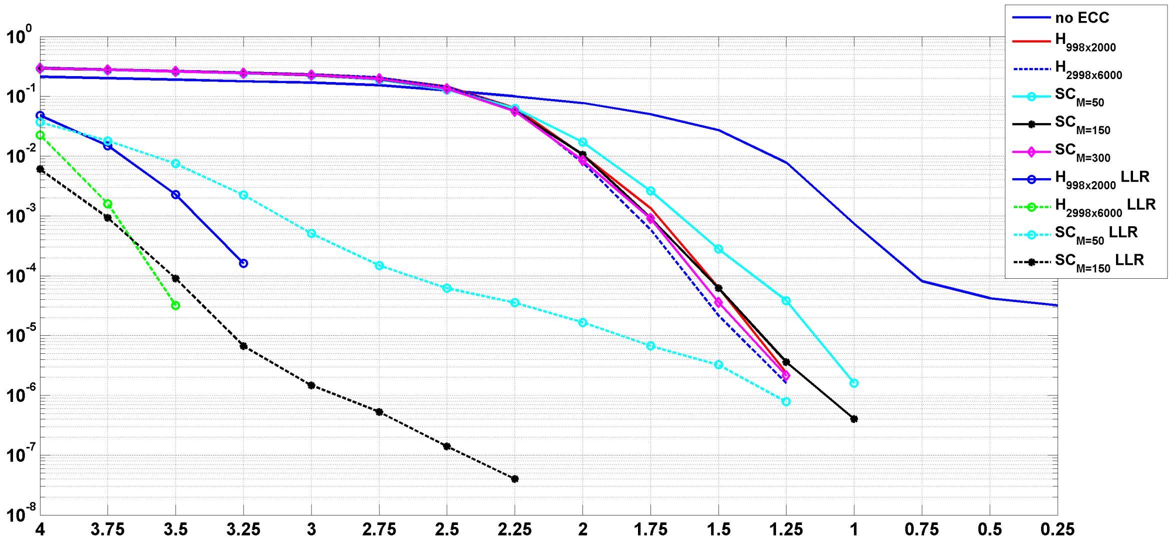

Figure 2.12. Different SER curves under various ECC schemes.

Note that, the flash channel output represents a non-binary symbol due to its MLC nature. Therefore, non-binary LLRs can be defined and employed in our analysis. However, for the sake of lower complexity (specifically in the ECC phase), it is preferred to continue in the binary LLR terminology. In our 4-level memory cell scenario, for each symbol two binary LLRs are calculated representing each of its binary bits. Let

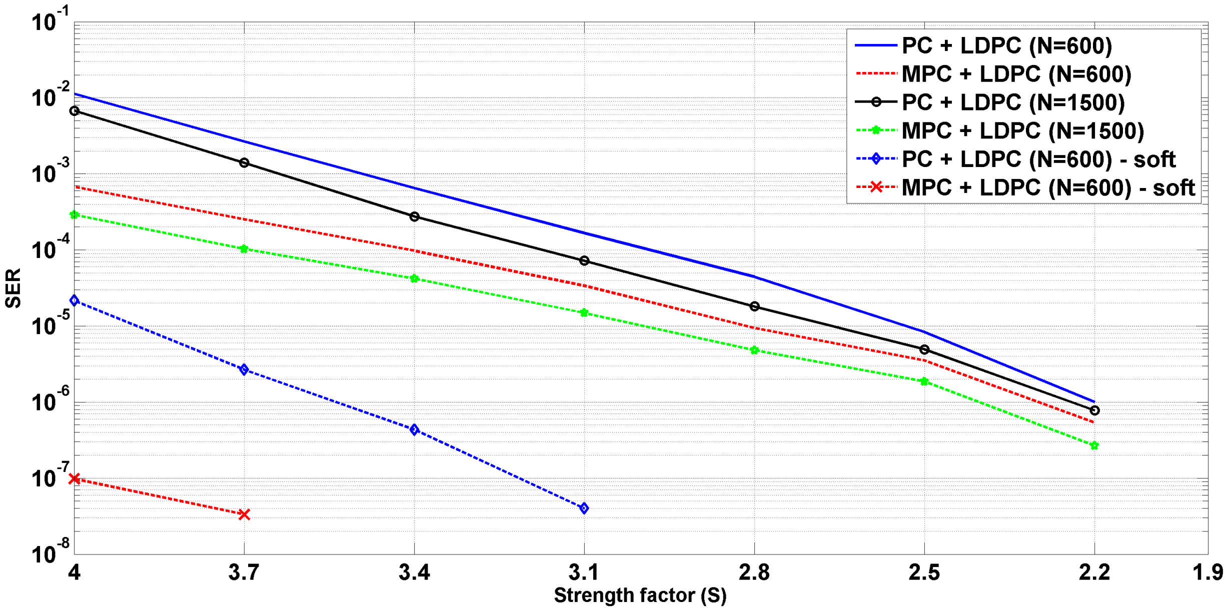

Figure 2.13. Different SER curves under various equalization-ECC schemes.

the binary tuple for a non-binary symbol be $s_i = (\alpha_{2i-1}\ \alpha_{2i})$, where $s_i \in \{0, 1, 2, 3\}$ and $\alpha \in \{0, 1\}$. LLR values are then calculated according to

$$\begin{aligned} l_{\alpha_{2i-1}} &= \log \frac{P(y_i|s_i = 0) + P(y_i|s_i = 1)}{P(y_i|s_i = 2) + P(y_i|s_i = 3)}, \\ l_{\alpha_{2i}} &= \log \frac{P(y_i|s_i = 0) + P(y_i|s_i = 2)}{P(y_i|s_i = 1) + P(y_i|s_i = 3)} \end{aligned} \tag{2.35}$$

As discussed earlier, probabilities $P(y_i|s_i)$ can be derived via extensive computer simulations or analytic Gaussian approximation approach.

The results for SER simulations are presented, in which two different ECC schemes, block LDPC and SC codes, have been employed. The parity-check matrices of two different block LDPC codes employed in our analysis are of sizes $\mathbf{H} \sim (998 \times 2000)$ and $\mathbf{H} \sim (2998 \times 6000)$, respectively. For SC codes, we have utilized three different realizations of the *ensemble B* with parameters $(L = 25, M = 50)$, $(L = 25, M = 150)$, and $(L = 25, M = 300)$, respectively. Number of decoding iterations (BP algorithm) is set to 100.

The curves in Figure 2.12 are produced via both hard and soft information input decoding, while channel LLRs are calculated according to Equation (2.35). As expected, soft input decoding considerably outperforms its hard information counterpart. Also, it is seen that larger code lengths result in better error performance.

To illustrate the error performance of the system, where both equalization and ECC are utilized, preliminary results in Figure 2.13 are presented. Equalization in based on the 2D hard-detection methods, and LDPC block code is chosen as the ECC scheme. It is seen that, the proposed low-complexity MPC equalizer can considerably improves the error rates. Furthermore, as presented earlier, soft-information based decoding crucially ameliorate the error rate of the system. However, extracting the soft information considering the latency requirements of the flash memory is still a challenging research area.

3. DISTRIBUTION OF THE THRESHOLD VOLTAGE

To compute the optimal detection thresholds and consequently estimate the error rate of the MLC flash channels, the time-varying PDF of the threshold voltage has to be accurately determined. However, due to the sophisticated two-dimensional channel characteristics and various complex error sources, there has not been any explicit channel modeling and derivation of the exact statistics of the flash channel [73].

Although few notable studies in the literature have been conducted to this end, they fail to capture the full essence of different distributions of various error sources simultaneously. For instance, [8] focuses on the retention noise analysis. In [9], effects of retention and RTN are not considered and a simple single-interferer ICI model is analyzed. Authors in [74] and [10] assume that the voltage distribution of each level follows the Gaussian PDF, while in [11], all noise sources, including ICI, are simply assumed to comply with Gaussian PDFs. Similar to analyses in the previous chapter, authors in [1] have assumed that the erroneous effect of all types of degradation, excluding ICI, can be represented by the zero-mean additive white Gaussian noise with a fixed variance. Moreover, the ICI definition has also been modified to ease the arithmetic tractability. All in all, an accurate and more comprehensive analysis of the distribution of the sensed voltage is still untouched and needs to be explored.

The similar analysis to the one in the previous chapter is considered here as well. However, a more complex hence accurate case is assumed, in which the erroneous effects of retention and error and the RTN, instead of the zero-mean fixed-variance additive Gaussian noise, are also taken into consideration. Therefore, the equation for the sensed voltage can be given by

$$y_{ij} = x_{ij} + z_{ij} + \nu_{r_{ij}} + \nu_{t_{ij}} \tag{3.1}$$

where $x_{ij}, i = \{1, \ldots, I\}, j = \{1, \ldots, J\}$, denotes the target voltage of the cell located on i^{th} wordline (row) and j^{th} bitline (column) of the block, representing non-binary symbol $s_{ij} \in \{0, 1, \ldots, L-1\}$. The ICI and retention error are represented by noise samples z_{ij} and $\nu_{r_{ij}}$, respectively, and $\nu_{t_{ij}}$ is the RTN sample. It is worth mentioning that all of these parameters are random variables drawn from different continuous distributions assumed to be mutually independent. As mentioned earlier in Section 2.1, the distribution of the L-level cell memory can be given as $f_X(x) = \sum_{i=0}^{L-1} \Omega_i f_{X_i}(x)$, where Ω_i denotes the weight of each individual PDF in the mixture. Corresponding mean and variance can be computed as [38]

$$\begin{aligned} \mu_x &= \sum_{i=0}^{L-1} \Omega_i \mu_i, \\ \sigma_x^2 &= \sum_{i=0}^{L-1} \Omega_i \sigma_i^2 + \sum_{i=0}^{L-1} \Omega_i (\mu_i - \mu_x)^2 \end{aligned} \tag{3.2}$$

Due to the equiprobability assumption on the input sequence, mixture weights are assumed, without loss of generality, to be $\Omega_i = \frac{1}{L}$. The following focus on the characteristics of the ICI, retention error, and RTN in detail.

3.1. Preliminaries

As mentioned previously, ICI originates from parasitic capacitance coupling effects among the surrounding cells, which makes the final threshold voltage of the target cell change while its interferer cells are being programmed [20]. Note that the target cell is affected by interferers, which are programmed afterwards. It is assumed that only immediate neighbors of a victim cell can affect the target cell, since the parasitic coupling quickly diminishes as the distance among cells grows.

Similar to [1] and [9], all-bitline programming has been utilized, in which all cells on a wordline are written simultaneously, and thus each cell generally suffers from several neighbor interferers located on the next row [28]. The general interference model of the victim cell is therefore given as

$$z_{ij} = \sum_{k=-\lfloor K/2 \rfloor}^{\lfloor K/2 \rfloor} (x_{i+1,j+k} - V_{e_{i+1,j+k}}) \gamma_{i+1,j+k} \tag{3.3}$$

where K represents the total number of interferers, $\lfloor\cdot\rfloor$ shows the floor function, and γ is the interference ratio, Each interferer show different level of strengths (γ) based on its distance to the target cell.

It is seen in Equation (3.3) that if the symbol stored in any of interfering cells is zero, no interference will be acted upon the victim cell, as its voltage does not change from the initial erased level. As mentioned, γ is assumed to comply with the truncated Gaussian distribution, as a limited number of neighbors on the next wordline is identified as interferers. Thus, $\gamma \sim \mathcal{TN}(\mu_\gamma, \sigma_\gamma^2, \omega)$, where ω determines the symmetric bound domain, and generally, σ_γ and ω are assumed to be proportional to μ_γ [17]. For the sake of comparison among a variety of ICI ranges, the latter values are scaled with ICI strength factor S.

Deriving the exact closed-form PDF of the interference in Equation (3.3) yields an analytically infeasible problem under the given setup, which originates from intractable integrals along with the presence of truncated Gaussian random variables. Thus, it is inevitable to resort to appropriate approximations to relieve the derivation problem.

As discussed, the total ICI consists of K components from different interferers on the next wordline. For an L-level flash memory, the effect of each interferer is either zero (zero-symbol programming generates no interference) or a voltage shift, which we assume to approximately follow the Gaussian distribution. This yields a mixture model for each of individual interferers as

$$f_{Z_i}(z) = \frac{1}{L}\delta(z) + \frac{(L-1)}{L\sqrt{2\pi\sigma_i^2}}\exp(\frac{-(z-\mu_i)^2}{2\sigma_i^2}) \tag{3.4}$$

where $i = \{1, \ldots, K\}$, $\delta(z)$ is the Dirac delta function, and for a dummy random variable X'

$$\begin{aligned} \mu_i &= (\mu_{x'} - \mu_e)\mu_{\gamma_i}, \\ \sigma_i^2 &= (\sigma_{x'}^2 + \sigma_e^2)\sigma_{\gamma_i}^2 + (\sigma_{x'}^2 + \sigma_e^2)\mu_{\gamma_i}^2 + (\mu_{x'}^2 - \mu_e^2)\sigma_{\gamma_i}^2 \end{aligned} \tag{3.5}$$

where distribution $f_{X'}$ is the mixture of $L-1$ uniform distributions, corresponding to programmed states, as the erased state is previously excluded and separately handled. Therefore, parameters $\mu_{x'}$ and $\sigma^2_{x'}$ can be computed according to Equation (3.2) as

$$\begin{aligned} \mu_{x'} &= \frac{1}{L-1}\sum_{i=1}^{L-1}\mu_{x'_i} = \frac{1}{L-1}\left(\sum_{i=1}^{L-1}V_{P_i} + \frac{(L-1)\Delta V}{2}\right), \\ \sigma^2_{x'} &= \frac{\Delta V^2}{12} + \frac{1}{L-1}\sum_{i=1}^{L-1}\left(\mu_{x'_i} - \mu_{x'}\right)^2 \end{aligned} \tag{3.6}$$

Consequently, the PDF of the total ICI can be written as

$$f_Z(z) = f_{Z_1}(z) * f_{Z_2}(z) * \ldots * f_{Z_K}(z) \tag{3.7}$$

where $*$ denotes the convolution operator. Plugging Equation (3.4) into Equation (3.7) and performing convolutions result in a mixture of Dirac delta function along with several Gaussian PDFs of different parameters and weights. The closed form expression can be written as

$$f_Z(z) = \sum_{j=1}^{2^K} \frac{w_j}{\sqrt{2\pi v_j}} \exp\left(-\frac{(z-m_j)^2}{2v_j}\right) \tag{3.8}$$

where w_j, m_j, and v_j are vector entries of $\mathbf{w}$, $\mathbf{m}$ and $\mathbf{v}$, respectively, as defined next. Note that for $j = 1$ in Equation (3.8), the mean and variance of the resulting Gaussian distribution equal to zero, meaning that it tends to the Dirac delta function.

To compute $\mathbf{w}$ according to the binomial theorem, let us define $u_h = \frac{(L-1)^h}{L^K}$, $h = \{0, 1, \ldots, K\}$. Then, weight vector $\mathbf{w}$ is built by sequentially repeating each element of u_h, $\binom{K}{h}$ times. For instance for $L = 4$ and $K = 3$, the weight vector is

$\mathbf{w} = \{\frac{1}{64}, \frac{3}{64}, \frac{3}{64}, \frac{3}{64}, \frac{9}{64}, \frac{9}{64}, \frac{9}{64}, \frac{27}{64}\}$. Furthermore,

$$\mathbf{m} = \mathbf{P} \begin{bmatrix} \mu_1 \\ \mu_2 \\ \vdots \\ \mu_K \end{bmatrix}, \; \mathbf{v} = \mathbf{P} \begin{bmatrix} \sigma_1^2 \\ \sigma_2^2 \\ \vdots \\ \sigma_K^2 \end{bmatrix} \tag{3.9}$$

where

$$\mathbf{P} = \begin{bmatrix} (0)_2 \\ (1)_2 \\ \vdots \\ (2^K - 1)_2 \end{bmatrix} \tag{3.10}$$

In other words, $\mathbf{P}$ is the matrix of size $2^K \times K$ containing all distinct binary vectors of length K.

As previously discussed, gradual and continual trapped charge leakage from the cell reduces the threshold voltage, and hence retained data severely degrades over time. The PDF of the data-dependent retention error, $f_{N_r}(\nu_r)$, can be modeled via a Gaussian, where its mean and standard deviation are given by [8, 75],

$$\mu_{r_i} = -C_r(V_{P_i} - \mu_e)N_{PE}^{\alpha}\left(\ln(1 + \frac{T_r}{T_{r_0}})\right), \quad \sigma_{r_i} = C_r'|\mu_{r_i}| \tag{3.11}$$

in which $i = \{1, 2, \ldots, L-1\}$ and C_r, C_r', and α are technology-related constants depending on the material properties and fabrication process of the memory cell. N_{PE} shows the number of PE cycles, T_r is the retention time, and T_{r_0} is the unit time set to one hour.

RTN originates from random fluctuations in the MOSFET's drain current or threshold voltage, where the PDF of the voltage fluctuation due to RTN has been empirically modeled as a symmetric exponential distribution centered around zero [36], i.e. zero-mean Laplace distribution with the PDF of

$$f_{N_t}(\nu_t) = \frac{1}{2\lambda} \exp\big(- \frac{|\nu_t|}{\lambda} \big) \tag{3.12}$$

where $\lambda = C_t N_{PE}^{\alpha}$ and C_t is also a material-related constant.

3.2. Error Probability Analysis

From Equation (3.1), we immediately realize that the PDF of the erroneously sensed voltages should follow

$$f_Y(y) = f_X(x) * f_Z(z) * f_{N_r}(\nu_r) * f_{N_t}(\nu_t) \tag{3.13}$$

Assuming the initial state is erased, the conditional PDF of the final voltage as a result of the convolutions can be shown to be of the form

$$\begin{aligned} f_{Y|S}(y|s=0) = \sum_{j=1}^{2^K} \frac{w_j}{4\lambda} \Bigg(& \exp\Big(\frac{2\lambda(y-(\mu_e+m_j))+(\sigma_e^2+v_j)}{2\lambda^2}\Big) \\ & \Big(\operatorname{erf}(\frac{\lambda(-y+(\mu_e+m_j))-(\sigma_e^2+v_j)}{\sqrt{2\lambda^2(\sigma_e^2+v_j)}})+1\Big) \\ & +\exp\Big(\frac{2\lambda(-y+(\mu_e+m_j))+(\sigma_e^2+v_j)}{2\lambda^2}\Big) \\ & \Big(1-\operatorname{erf}(\frac{\lambda(-y+(\mu_e+m_j))+(\sigma_e^2+v_j)}{\sqrt{2\lambda^2(\sigma_e^2+v_j)}})\Big)\Bigg) \end{aligned} \tag{3.14}$$

where $\operatorname{erf}(\xi) = \frac{2}{\sqrt{\pi}} \int_0^{\xi} \exp(-\tau^2)\, d\tau$ is the error function.

To derive PDFs conditioned on other potential inputs, i.e., $f_{Y|S}(y|s=i)$, $i = \{1, \ldots, L-1\}$, a more complex convolution with the inclusion of the uniform distribution has to be computed. This can be solved via integration by parts. Let us define the function given in Equation (3.15). Then, we have $f_{Y|S}(y|s=i) =$

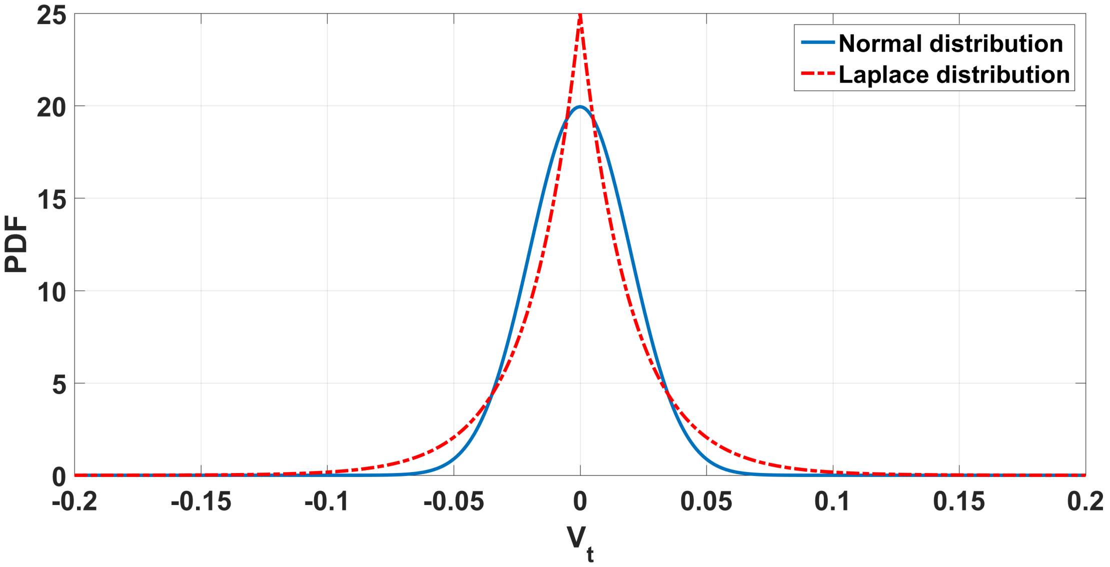

Figure 3.1. An illustration of the PDF of zero-mean normal and Laplace distributions, where $\lambda = 0.02$.

$h(y, \Delta V, \eta_i, \theta_i) - h(y, 0, \eta_i, \theta_i)$, where $\eta_i = (V_{P_i} + \mu_{r_i})$ and $\theta_i = (\sigma_{r_i}^2 + \lambda^2)$, respectively.

$$\begin{aligned} h(y, t, \eta_i, \theta_i) = \sum_{j=1}^{2^K} \frac{w_j \exp\left(-\frac{t+y+\eta_i+m_j}{\lambda}\right)}{4\Delta V} & \Bigg(\exp\left(\frac{4\lambda\,(t+\eta_i+m_j)+\theta_i+v_j}{2\lambda^2}\right) \\ & \left(1 - \operatorname{erf}\left(\frac{\lambda\,(t-y+\eta_i+m_j)+\theta_i+v_j}{\lambda\sqrt{2(\theta_i+v_j)}}\right)\right) \\ & - \exp\left(\frac{4\lambda y+\theta_i+v_j}{2\lambda^2}\right)\left(\operatorname{erf}\left(\frac{\lambda\,(t-y+\eta_i+m_j)-\theta_i-v_j}{\lambda\sqrt{2(\theta_i+v_j)}}\right)+1\right) \\ & + 2\exp\left(\frac{t+y+\eta_i+m_j}{\lambda}\right)\operatorname{erf}\left(\frac{t-y+\eta_i+m_j}{\sqrt{2(\theta_i+v_j)}}\right)\Bigg) \end{aligned} \tag{3.15}$$

It is important to note that derived conditional probabilities above may not yield stable results for very small values of λ (e.g., in the vicinity of $\lambda = 0.01$), which is likely for a large portion of the flash's lifespan according to Equation (3.12). Therefore for simplicity and to relieve the problem under small values of λ, we assume that the zero-mean Laplace distribution is approximated with the reminiscent zero-mean Gaussian distribution of the standard deviation of λ, which is depicted in Figure 3.1. This modification is justified as for smaller λ values, both of theses symmetric distributions approach the Dirac delta function. Thus, Equation (3.14) becomes

$$f_{Y|S}(y|s=0) = \left(\sum_{j=1}^{2^K} \frac{w_j}{\sqrt{2\pi(\sigma_e^2+v_j+\lambda^2)}} \exp\Big(-\frac{(y-(\mu_e+m_j))^2}{2(\sigma_e^2+v_j+\lambda^2)}\Big)\right) \tag{3.16}$$

The rest of the conditional probabilities are also simplified to

$$f_{Y|S}(y|s=i) = \sum_{j=1}^{2^K} \frac{w_j}{2\Delta V}\left(\text{erf}\Big(\frac{y-\eta_i-m_j}{\sqrt{2(\theta_i+v_j)}}\Big) - \text{erf}\Big(\frac{y-(\Delta V+\eta_i+m_j)}{\sqrt{2(\theta_i+v_j)}}\Big)\right) \tag{3.17}$$

where $i = \{1, 2, \ldots, L-1\}$.

Read (detection) thresholds $\tau = [\tau_1\ \tau_2 \ldots \tau_{L-1}]$ and corresponding domains are then simply computed numerically by finding the intersection points of these L PDFs. Therefore, the general probability of symbol error can be written as

$$\begin{aligned} \text{P(e)} = \sum_{i=0}^{L-1} \text{P}(e|s=i)P(s=i) = \frac{1}{L}\Bigg(\int\limits_{y\notin(-\infty,\tau_1)} f(y|s=0)dy \\ + \int\limits_{y\notin(\tau_1,\tau_2)} f(y|s=1)dy + \ \ldots\ + \int\limits_{y\notin(\tau_{L-1},\infty)} f(y|s=L-1)dy\Bigg) \end{aligned} \tag{3.18}$$

Integrations of the conditional PDFs in Equation (3.16) and Equation (3.17) can be evaluated analytically by the technique of integration by parts, resulting in closed-form expressions given below.

$$\int f(y|s=0)\,dy = \sum_{j=1}^{2^K} \frac{w_j}{2}\,\text{erf}\left(\frac{y-(\mu_e+m_j)}{\sqrt{2(\sigma_e^2+v_j+\lambda^2)}}\right) + C_0 \tag{3.19}$$

Table 3.1. Simulation parameters for the general case.

Parameter	Value	Parameter	Value
V_P	[2.55, 3.15, 3.75] volts	μ_{γ_V}	0.08S
ΔV	0.3	σ_{γ_V}	$0.3\mu_{\gamma_V}$
μ_e , σ_e	1.1 , 0.35	ω_{γ_V}	$0.2\mu_{\gamma_V}$
I , J	128 , 2×10^4	μ_{γ_D}	0.006S
C_r	2.35×10^{-4}	σ_{γ_D}	$0.3\mu_{\gamma_D}$
$C_r^{'}$	0.4	ω_{γ_D}	$0.2\mu_{\gamma_D}$
C_t	2.52×10^{-4}	α	0.62

$$\begin{aligned} \int f(y|s=i)\, dy = \sum_{j=1}^{2^K} \frac{w_j}{2\Delta V} \Bigg((y - \eta_i - m_j)\, \mathrm{erf}\left(\frac{y - \eta_i - m_j}{\sqrt{2(\theta_i + v_j)}}\right) \\ + \sqrt{\frac{2(\theta_i + v_j)}{\pi}}\, \mathrm{e}^{\frac{-(y-\eta_i-m_j)^2}{2(\theta_i+v_j)}} - \Big((y - \eta_i - m_j - \Delta V) \\ \mathrm{erf}\left(\frac{y - \eta_i - m_j - \Delta V}{\sqrt{2(\theta_i + v_j)}}\right) + \sqrt{\frac{2(\theta_i + v_j)}{\pi}}\, \mathrm{e}^{\frac{-(y-\eta_i-m_j-\Delta V)^2}{2(\theta_i+v_j)^2}} \Big) \Bigg) + C_i \end{aligned} \tag{3.20}$$

where $i = \{1, 2, \ldots, L-1\}$ and constants $C_0, C_i \in \mathbb{Z}^+$.

3.3. Results and Discussions

Presented simulation results in this section are based on the 4-level MLC flash memory channel ($L = 4$), where similar to [9], the number of interferes is set to $K = 3$, namely one vertical (V) and two diagonal (D) interferers. Parameters employed in our computer simulations, also used in [17, 75], are listed in Table 3.1.

Figures 3.2 and 3.3 present four conditional PDFs calculated through computer simulations along with the evaluation of their analytically-driven counterparts under two different sets of channel parameters. It is observed that the analytical approach (based on aforementioned approximations) provides an extremely close match to the underlying simulated PDFs, in which distributions of γ and RTN are simulated to

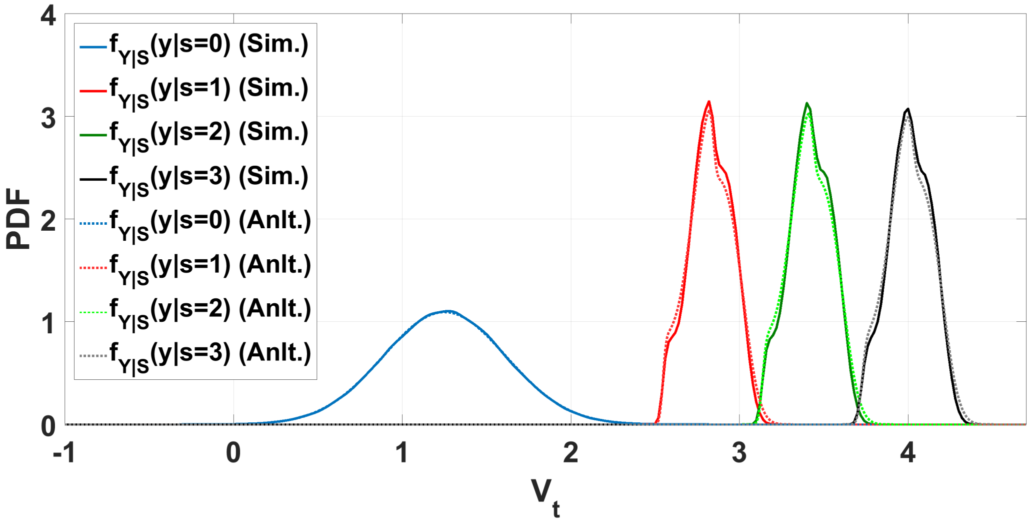

Figure 3.2. Conditional PDFs of the sensed voltage derived from simulations (Sim.) and described analytical (Anlt.) process, where $S = 1$, $N_{PE} = 100$, $T_r = 50$.

follow the truncated Gaussian and Laplace distributions, respectively. For the sake of comparison, resultant PDFs are compared to ones in [1], in which a similar setup for channel modeling has been assumed, and results are shown in Figures 3.4 and 3.5. It is clearly observed that in the latter model, in which the ICI and aforesaid noises have been modified for tractability purposes, they largely deviates from the underlying realistic distributions.

Moreover, Figure 3.6 shows the simulated and aforementioned analytical SER evaluations under different channel parameters for a range of ICI strengths (S). As can be seen, the corresponding analytical results for error rates closely follow the simulated SERs, which confirms the accuracy and importance of the derived analytical PDF and SER expressions.

To demonstrate the robustness and reliability of derived expressions, another set of SER experiments has been carried out, in which ICI ratios of the channel in Table 3.1, have been aggravated to $\mu_{\gamma_V} = 0.12S$ and $\mu_{\gamma_D} = 0.01S$, i.e., the interference is artificially amplified. The resulting SER curves are presented in Figure 3.7, and as can immediately be noticed, SER values have been degraded compared to the Figure 3.6

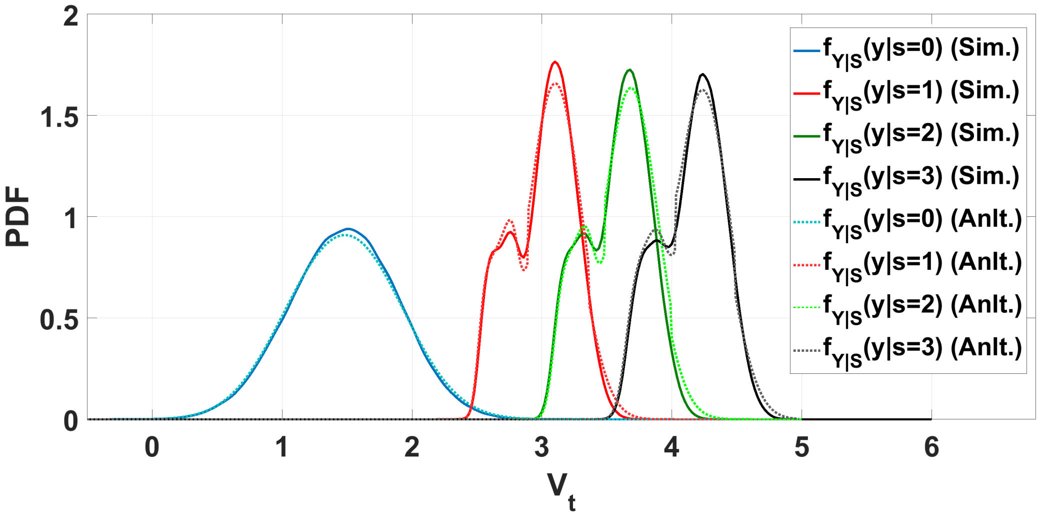

Figure 3.3. Conditional PDFs of the sensed voltage derived from simulations (Sim.) and described analytical (Anlt.) process, where $S = 2.5$, $N_{PE} = 500$, $T_r = 100$.

due to a more severe interference. However, simulated and theoretic SER expressions are still in a very close match.

Altogether, deriving an accurate PDF of the sensed threshold voltage is essential for precisely reading off the cell values and implementing error control coding schemes. Previous studies did not consider the full integration of various dominant error sources and their realistic characterizations. However in this work via accurate approximations, generalized closed-form analytic expressions for conditional PDFs of the sensed voltages and system's error endurance performance are presented. Computer simulations of both PDFs and SERs confirm that the derived formulas substantially match with the underlying PDFs and can be employed in further theoretical analysis and practical development of the MLC flash channel.

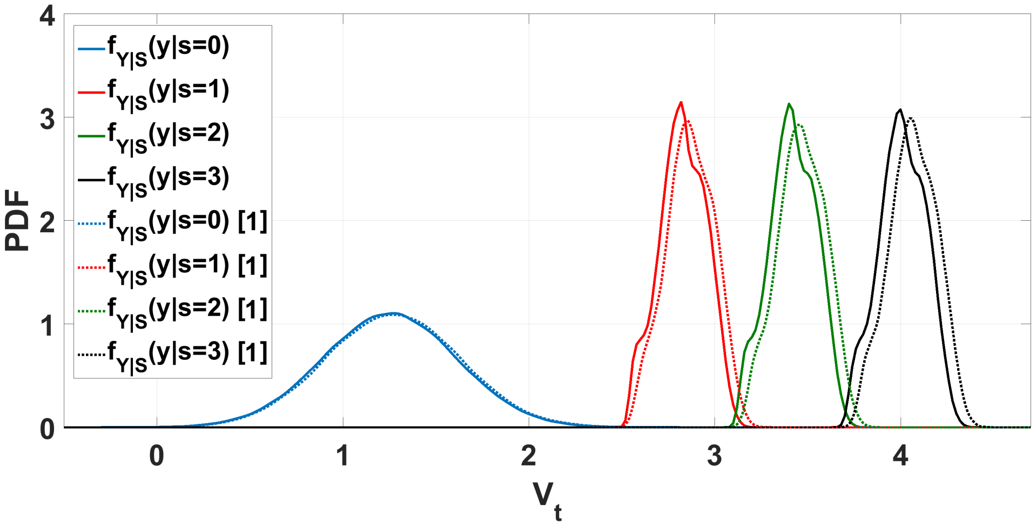

Figure 3.4. Conditional PDFs of the sensed voltage derived from our simulations and the corresponding model in [1], where $S = 1$, $N_{PE} = 100$, $T_r = 50$.

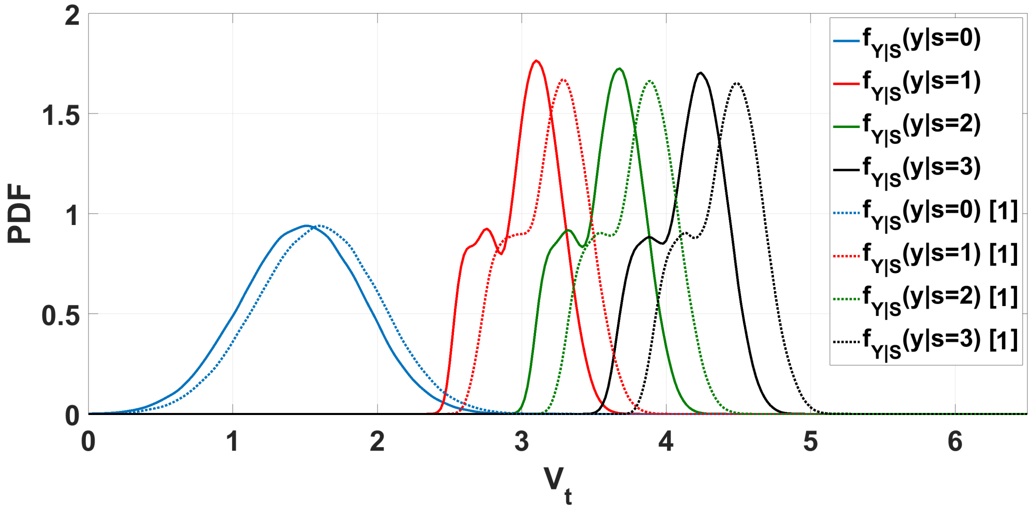

Figure 3.5. Conditional PDFs of the sensed voltage derived from our simulations and the corresponding model in [1], where $S = 2.5$, $N_{PE} = 500$, $T_r = 100$.

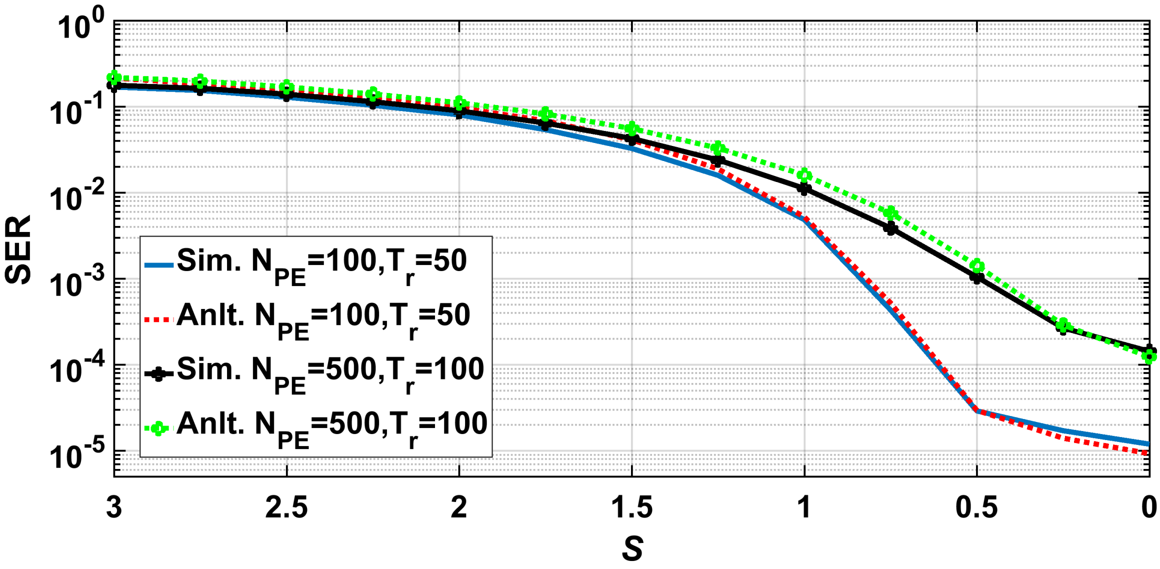

Figure 3.6. Analytical and simulation based SER of the flash channel under different parameters.

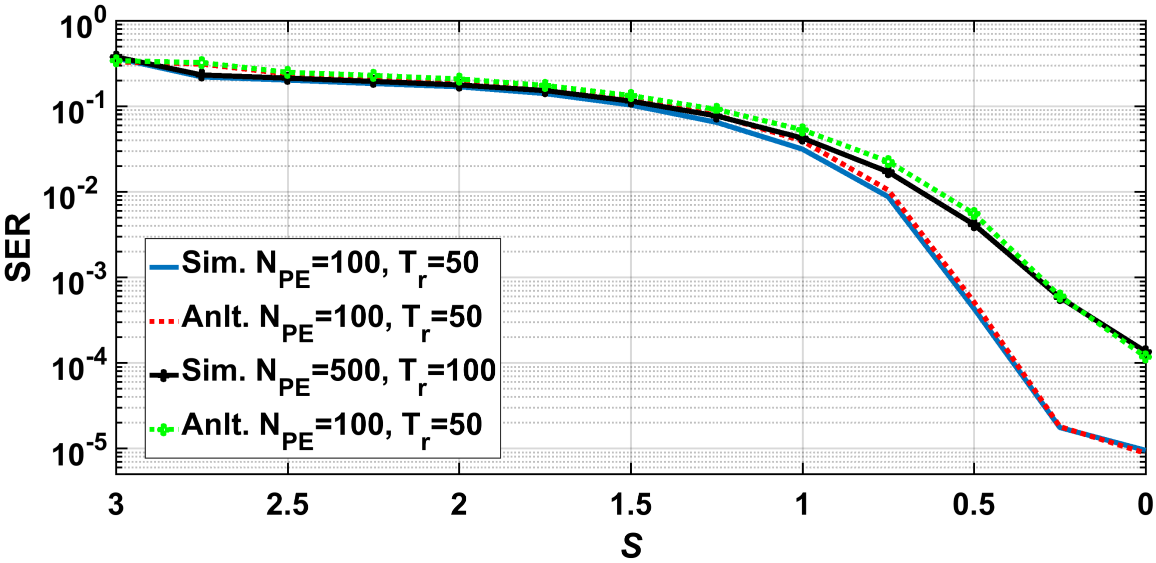

Figure 3.7. Analytical and simulation based SER of the aggravated channel under different parameters.

4. A MACHINE LEARNING APPROACH

Machine learning attempts to teaches machines and computers to operate in way that seems natural to humans, i.e. learning from previous experiences. Machine learning algorithms use computational methods to learn information directly from the given data without relying on predetermined models and descriptive equations. The algorithms adaptively improve their performance as the number of samples available for learning increases. Machine learning algorithms find natural patterns in data that generate insight to improve the decision making and predictions. The goal of machine learning is to program machines and computers to solve a given problem via optimizing a performance criterion using example data and past experience, which has already been successfully utilized in many applications including medical diagnosis, stock trading, energy forecasting, video and language processing, and marketing and advertisement [76]. The root of machine learning lies in the statistics, however it complements statistical approaches by focusing on algorithmic issues, computational efficiency, and data engineering. It is important to note that, machine learning is utilized in the sort of problems, where robust solutions cannot be developed through pure rule-based mathematical-statistical techniques. There are enormous problems, in which the final solution depends on a large number of governing factors, and therefore due to the complexity of accurately tracking and modeling these factors and their inter-relations, the problem indeed becomes infeasible. Another issue is the large scale of some problems, which results in a tedious and inefficient statistical model building. Therefore, in such scenarios, employing machine learning techniques is an advantageous option [77].

The learning process has two steps, training and testing. In the former phase, available samples are input to the machine and parameters are learned and the learning model is constructed. In the test phase, the learned model is employed to perform prediction on the new data set. Machine learning algorithms are divided into two types: supervised learning, which trains a model on known input and output data so that it can predict future outputs. In other words, during the training phase, sample observations and their corresponding labels are input to the learning machine. In unsupervised learning however, only the sample observations are given to the machine and it finds hidden patterns or intrinsic structures in input data. An important application of supervised learning is the data classification, in which the goal is to classify observations into the given set of distinct categories, while clustering is the main functionality of the unsupervised learning.

It was inferred from previous discussions that the derived channel model is non-stationary and its characteristics may drastically change with number of usage (PE cycles), retention time, and operation/environment temperature. Although the corresponding information of some of these factors, e.g. the number of PE cycle, may be tracked and recorded by the controller section of some advanced devices to be utilized in channel model adaptations, some controlling data are impractical to be perpetually collected and utilized. Therefore, relying solely on the mathematical/probabilistic channel model derived in previous section to accurately specify the decision boundary regions, even though it represents a very accurate model with respect to PDF derivations, cannot guarantee a reliable error performance due to the non-stationarity. Thus, as the main contribution of this chapter, we focus on the adaptive estimation of the channel parameters through application of machine learning in the aforementioned scenario.

Learning process can be divided into two main groups: supervised and unsupervised learning. In supervised learning, in contrast to the unsupervised approach, it is assumed that some prior knowledge on inputs and their corresponding outputs (labels) are available, from which machine learning tool initially tries to learn the underlying relation. This step is referred to as the training phase. Next, in the test (prediction) phase, the learned relation is applied on new observations to complete the process.

4.1. Kernel Density Estimation

As discussed in previous chapters, various error sources degrade the MLC memory performance, including ICI, retention error, and RTN. Accurate integration of these error sources into the analytical model to optimally derive the governing probability distributions and consequently the detection thresholds to minimize error rates lie at the heart of MLC research. Nevertheless, utilizing static derivations will not be addressing the detection problem through the entire memory's lifespan, as aforementioned error sources exhibit a strong non-stationary behavior that is correlated to the data retention time and the utilization patterns of the flash memory. In this section, a novel low-complexity implementation of an non-parametric learning mechanism, kernel density estimation, shall be used to periodically estimate the underlying probability distributions and hence approximate the optimal detection performance for time-varying all-bitline MLC flash channel.

In this section, an alternative approach is proposed to extract the underlying channel model and its necessary information to be utilized in the corresponding signal processing systems via a learning mechanism based on the kernel density estimation (KDE). In this technique, we periodically learn the data distribution from a finite set of training samples. This learning process results in the dynamic estimation of conditional probabilities necessary for near-optimal sequence detection of the stored data values. This periodic estimation process provides adaptivity and approximation to the non-stationary nature of the flash channel which is not addressed by previous literature.

Given the sample observation (training) data, estimating the unknown underlying probability density function is an important class of statistical problems. An elementary approach as the foundations of a variety of estimation procedures is the histogram-based density estimation, which is not a suitable choice in many of practical scenarios. The main disadvantage of histograms is the discrete nature of the plot, in which for a fixed number of observations, several bins may end up with no observation, ultimately resulting in a bumpy and discontinuous density estimate. The most common approach however, to produce a more smooth probability estimate, is the use of non-parametric kernel density estimation [78]. KDE is a nonparametric density estimator, which requires no prior knowledge and assumption of the underlying density function or the family it might belong to. KDE automatically learns the shape of the density from the data, and due to this nonparametric nature of the KDE, it is a very popular approach for data drawn from an unknown or complicated distribution.

Let $\{x_1, x_2, \cdots x_n\}$ be independent and identically distributed random variables drawn from an unknown distribution f. The 1D KDE of the distribution f can be expressed as

$$\hat{f}_n(x) = \frac{1}{nh} \sum_{i=1}^{n} \mathcal{K}(\frac{x - x_i}{h}) \tag{4.1}$$

where h is called the smoothing bandwidth and $\mathcal{K}$ is the kernel function. Therefore, direct evaluation of Equation (4.1) at m evaluation points, which is governed by the sensing resolution, for n data sample involves $\mathcal{O}(mn)$ kernel evaluations and $\mathcal{O}(mn)$ multiplications and additions [79]. Therefore, to reduce the implementation complexity, the size of the training set and also quantization level of the threshold voltage sensing should decrease, which clearly results in a poorer density estimation.

For problems, in which the underlying distribution is conjectured to follow (or close to) the Gaussian form, Gaussian kernel (zero-mean and unit variance) is chosen. Note that however, selecting other common types of kernels do not significantly degrade the resulting density estimate [80]. In contrast to the kernel selection, bandwidth choice, similar to the bin width choice in histograms, is crucial and has been a widely-addressed problem in the relevant literature, and yet no procedure has been considered universally the best. A very small bandwidth produces too detailed curvature and thus causes an estimation with small bias and large variance. In contrast, larger h values result in a low variance at the cost of larger bias. The mean integrated square error (MISE), similar in spirit to the mean square error, is the most applied error measurement for the KDE and is defined as $\int \mathrm{E}\left(\hat{f}_n(x) - f(x)\right)^2 dx$. The optimal bandwidth minimizing the MISE can be mathematically obtained, however in practical computations, another useful distance measure between the estimator and the true density is used, which is asymptotically the same as the MISE, asymptotic mean integrated squared error (AMISE) defined as [81, 82]

$$AMISE = \sigma_k^4 h^4 \frac{R(f^2)}{4} + \frac{R(\mathcal{K})}{nh} \tag{4.2}$$

where

$$\begin{aligned} R(L) &= \int L^2(x)dx, \\ \sigma_k^2 &= \int x^2 \mathcal{K}(x)dx \end{aligned} \tag{4.3}$$

Minimizing the AMISE, results in

$$h = \left(\frac{R(\mathcal{K})}{n\sigma_k^4 R(f^2)} \right)^{\frac{1}{5}} \tag{4.4}$$

Note that, the resulted optimal bandwidth value is impractical to use, since it is still a function of the unknown distribution f through $R(f^2)$. Therefore to solve the problem, the unknown distribution can be replaced with a known reference distribution. For instance, in the general case of (close to) Gaussian assumption of the underlying density and kernel function, a very low-complexity rule-of-thumb for the optimal bandwidth, referred to as the Silverman's rule-of-thumb, which is adaptive to the underlying data through its variance, is given as [78, 80]

$$h = 1.06\hat{\sigma}n^{-0.2} \tag{4.5}$$

where $\hat{\sigma}$ is the sample standard deviation of the training data.

In Figures 4.1 and 4.2, effects of the data length and the bandwidth on the performance of the KDE are shown, where the underlying PDF is a normal distribution with mean and standard deviation of 1 and 0.5, respectively. As can be seen, larger sample sizes result in a smoother density estimation. Furthermore, it is clearly inferred that the bandwidth selection is vital, as larger h values produce over-smoothed estimates, while very smaller h values render intolerably under-smooth probability densities.

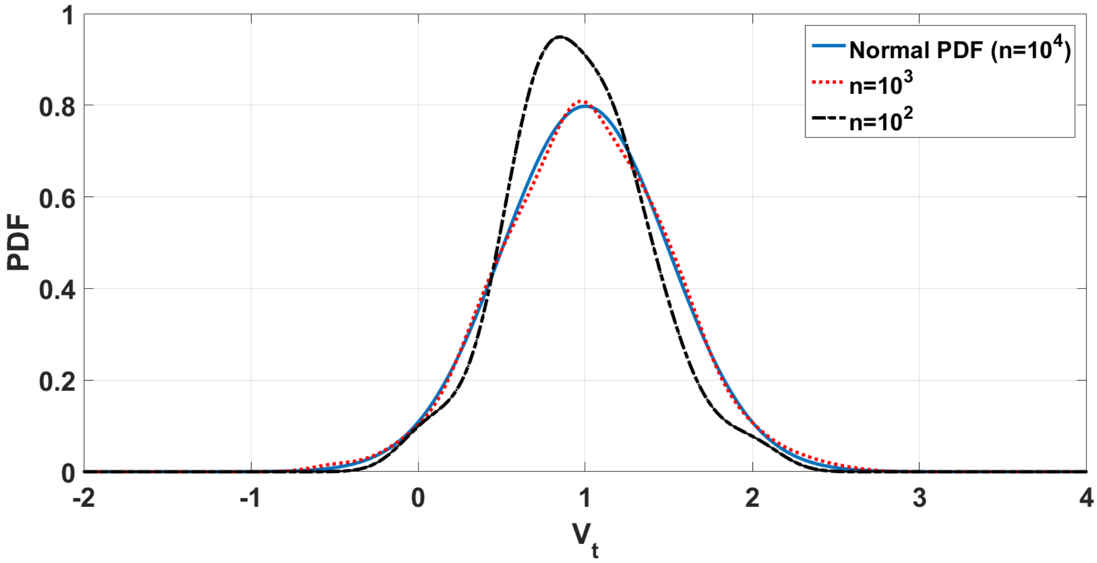

Figure 4.1. Illustration of the data length on the performance of the KDE.

In our flash channel model, all the required conditional probabilities, which are used to compute the thresholds and error rates, are calculated via the KDE method. For instance, for $l = 2$-bit MLC flash channel, we have sixteen conditional probabilities employed in the sequence detection method that shall be derived via the KDE on the provided training data samples in our simulations. More specifically, to derive the probability $P(y_i|s_i s_{i-1}),\ i = 1, 2, \ldots, I$, one has to access prior data symbols on two pages (rows), and y_i i.e., the channel output.

4.1.1. Results and Discussions

Accurate analytical analysis of the flash channel including deriving necessary PDFs once for all and optimal detection thresholds require real time access to governing parameters of the channel. For some parameters, e.g., the PE cycle, this might be realized, however it is impractical for the rest. Therefore, extracting the underlying PDFs from a set of available training data renders an appropriate approach. However, this approach comes at the cost of extra computational burden and delay due to processing. Therefore, we seek an acceptable trade-off between the error performance of the flash channel versus the overall computational complexity and delay.

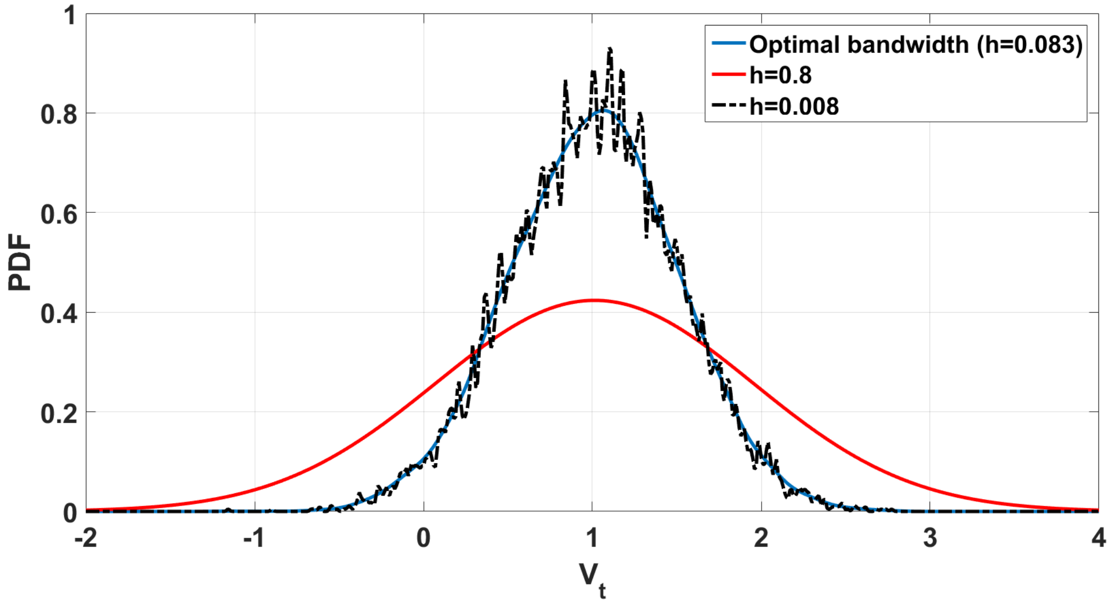

Figure 4.2. Illustration of the bandwidth selection on the performance of the KDE.

Note that, the list of employed values for different parameters in our computer simulations are shown in Table 3.1. From discussion and computer simulation results presented in the Chapter 2, the inferior performance of the symbol-by-symbol detection/equalization has been be clearly observed, particularly at high ICI ratios (large S). Therefore, experiments in this section only consider the Viterbi-like sequence detection/equalization method for achieving a better error performance.

Figure 4.3 shows the SER curves of the sequence detection under different sensing resolutions. It is clearly seen that higher sensing precision, e.g. 8 bits, clearly improve the error performance. In order to obtain Q bits of reading resolution, a flash cell has to be read Q times, each time with a different threshold voltage. However, it results in a higher sensing latency, which may be intolerable for the overall utility of the flash device, as the throughput of storage systems and its data transfer speed is limited. On the other hand, each round of reading, produces higher level of soft information, which is necessary for a modern and better equalization and ECC schemes. Therefore, establishing a optimized fashion in the read latency for the soft information generation is still a major focus point of the flash memory's research community [9]. Note that, for the rest of computer simulations to show the potentials of the proposed intelligent

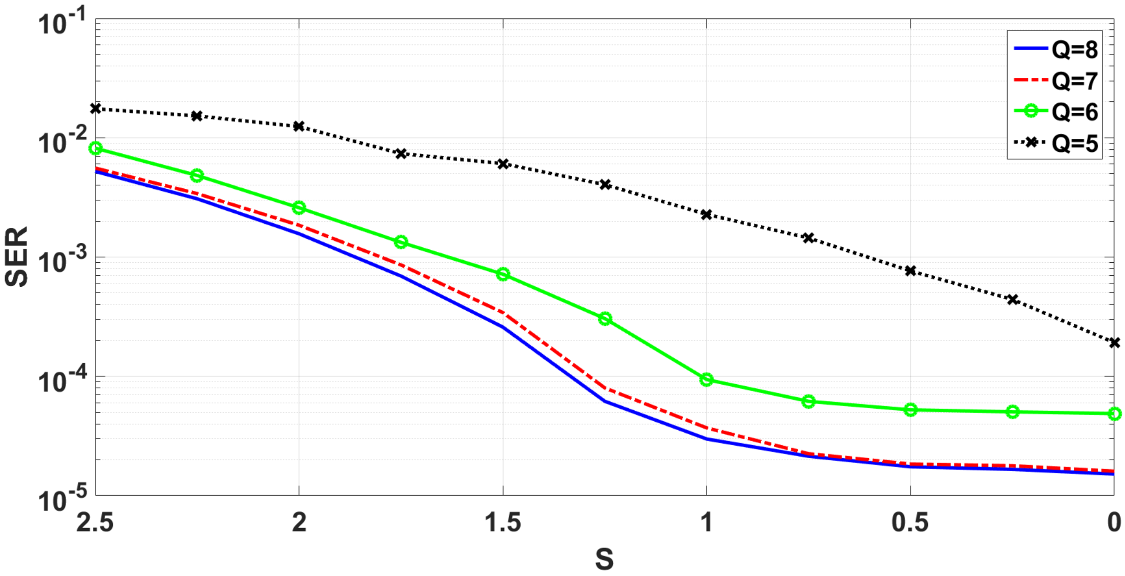

Figure 4.3. SER of the sequence detection equalization under different sensing resolutions, where $N_{PE} = 100$ and $T_r = 50$.

method, sensing resolution is assumed to be Q=8 bits.

To demonstrate the error performance degradation utilizing the static prior channel information (offline training) (e.g., conditional PDFs), we assume a scenario in which the prior information is derived at a fixed condition (for instance and without loss of generality $N_{PE} = 100$ and $T_r = 50$) and is repeatedly used throughout a large portion of the memory usage, i.e., N_{PE} is increasing under a fixed retention time assumption. Note that, the proposed periodic update scheme of the channel information can be extended to more realistic cases, in which the retention time fluctuates. However for deriving preliminary results, retention time is simply assumed to be fixed. The family of curves corresponding to different N_{PE} values are illustrated in Figure 4.4, which clearly justifies the advantage of the periodic update of the channel information (online training).

Establishing a good trade-off between the error performance and the system's constraints via an optimal training interval is key to the design. To decrease longer overheads, redundancies, and KDE's complexity, we can decrease the training length

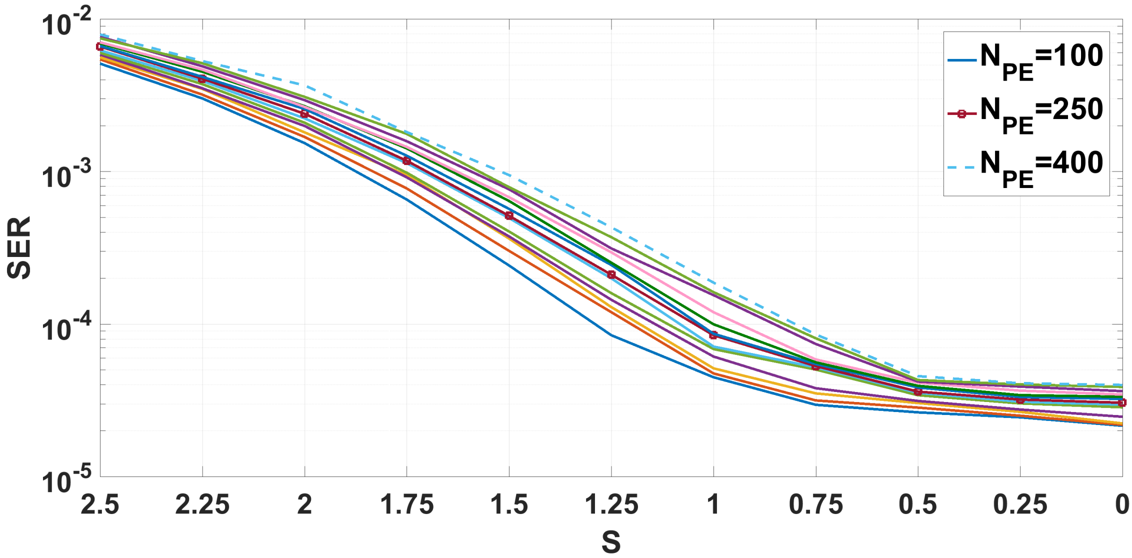

Figure 4.4. SER of the sequence detection equalization utilizing offline training under different N_{PE} values, i.e., $N_{PE} = 100, 125, 150, \ldots, 400$, where $T_r = 50$. Note that, to avoid confusion, only three legends are shown.

(TL), which is illustrated in Figure 4.5. TL of 8000 symbols means an approximate data redundancy of 0.781% that shrinks to 0.048% for a TL of 500 symbols. However as can be seen from the figure, utilizing a training set with large TL (=8000) gives around half an order better SER than the short overheads such as TL=500.

The second parameter to control the system's computational complexity as well as reliability is the training interval (TI) selection. In other words, a reasonable scheme should lie between the offline and fully online training schemes, in which for every P/E cycle, the channel is learned and PDFs are updated. To analyze this case, the channel at a fixed ICI strength $S = 1.5$ is selected and the SER is evaluated under four different cases, as given in Fig 4.6. Note that this specific value of ICI strength (and lower values) results in an equalized SER performance below 10^{-3}, which is an acceptable rule of thumb input error rate for reliable error correction decoding. It is observed that read errors increase by the N_{PE}, as expected. However the discontinuities in performance curves, i.e., where the SER curve suddenly drops, show the specific set of PE cycles in which the training is performed, PDF tables are subsequently updated, and therefore SER performance is pulled down. It is observed that periodic updates, at even distant

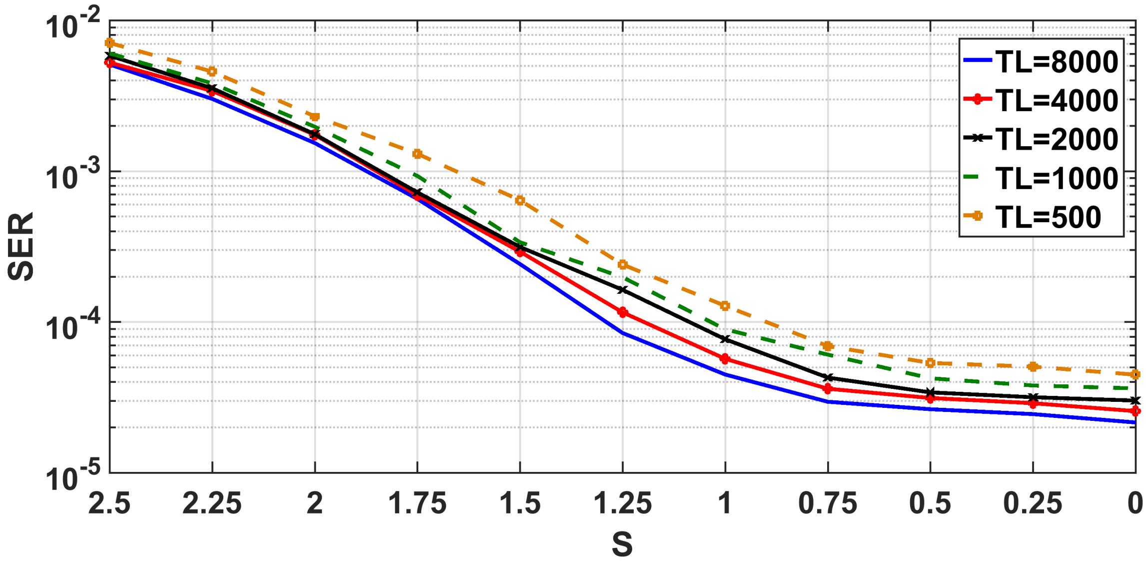

Figure 4.5. SER of the sequence detection equalization under different training lengths, where $N_{PE} = 100$ and $T_r = 50$.

PE cycles, can considerably improve the SER performance compared to the offline prior information. This result clearly demonstrates how important the proposed adaptive scheme could be for maintaining a reliable operation and ensuring the longevity of the MLC flash drives.

In conclusion, packing more bits into a given MLC memory cell leads to various error types which are hard to model, analyze, and combat against using conventional techniques. Thus, a periodic learning technique based on KDE to adapt to non-stationary all-bitline MLC flash channel is proposed, in which the system periodically learns the dynamically changing statistical properties of the channel from a finite set of training samples to be used in the near-optimal sequence detection of stored data. This periodic estimation and detection scheme can prolong the lifetime of the MLC flash drive or for a given lifetime, it can allow adaptive rate error control coding and hence allow more raw user data to store.

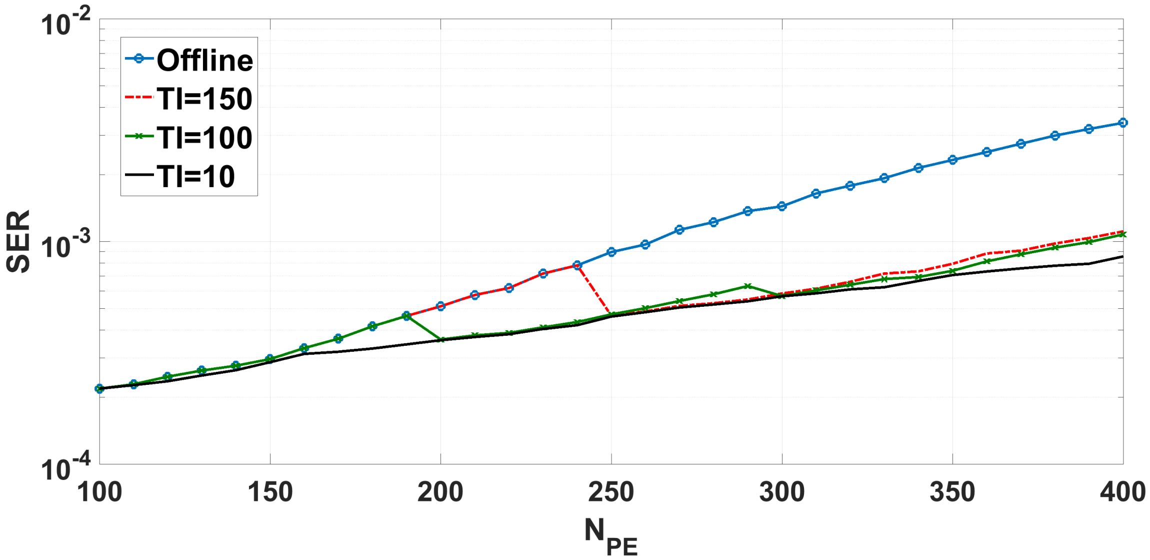

Figure 4.6. SER of the sequence detection equalization under different training intervals, where $S = 1.5$ and $T_r = 50$.

4.2. Support Vector Machine

In this section, the basic concept of support vector machines (SVM) is reviewed and later the potential aspects of its implementation in the flash memories to tackle the non-stationary nature of corresponding error sources are discussed. Let us first assume that, there is a large or perhaps infinite number of observations or patterns, which should be classified into several classes. The main problem is that there is no known algorithm or rule, possibly due to the very complex nature of the problem, to perform the classification task. However, a finite sample of observations with known class labels are available (training dataset). The goal is to teach the classification system by the available labeled observations, so that it learns the underlying rule to classify the remaining observations of the set (test dataset), while achieving the the minimum classification error.

SVM is a powerful supervised machine-learning-based binary classification algorithm, i.e. separating the data set into two distinct classes. Note that, the idea can be extended to multicalss problems, which will also be discussed. The fundamental idea behind SVM is to find the hyperplane, which linearly separates data points of two

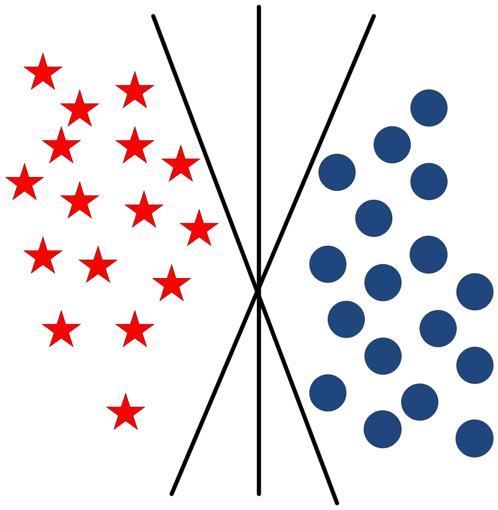

Figure 4.7. An illustration of the maximal margin classification.

classes with the maximal margin [83]. Thus, SVM performs the optimization by simultaneously minimizing the classification error and maximizing the geometric margin between the classes. The points located on the margin of the separating hyperplane are the support vectors specifying the decision function.

Beginning with linearly-separable SVM, let $\{(\mathbf{x}_1, \mathsf{y}_1), (\mathbf{x}_2, \mathsf{y}_2), \ldots, (\mathbf{x}_n, \mathsf{y}_n)\}$ be the training dataset, where $\mathbf{x}_i \in \mathbb{R}^d$ is the multi-dimensional observations and $\mathsf{y}_i \in \{-1, +1\}$ represents labels of two classes. Therefore based on available training data features $\mathbf{x} \in \mathbb{R}^{d\times n}$, the goal is to classify given test observations $\mathbf{x}' \in \mathbb{R}^{d\times m}$ into one of the binary classless $+1$ or -1 by finding a separating hyperplane with the largest margin possible. The margin of the hyperplane is defined as the shortest distance between the positive and negative samples closest to the hyperplane illustrated in Figure 4.7. As can be seen, there might be infinite hyperplanes that perfectly separates data points of two classes, but only one hyperplane guarantees the optimal margin. Assume that all points in the vector $\mathbf{x}$ lie exactly on the separation hyperplane, therefore $\mathbf{w}^T\mathbf{x} + b = 0$, where $\mathbf{w}$ is the normal weight vector to the hyperplane and b is the interceptor (bias) term. Therefore, any data point above the decision boundary should belong to the class +1, and similarly points below the separating hyperplane belong to the class -1.

$$\begin{aligned} \mathbf{w}^T\mathbf{x_i} + b > 0, &\quad \text{for } \mathsf{y}_i = +1, \\ \mathbf{w}^T\mathbf{x_i} + b < 0, &\quad \text{for } \mathsf{y}_i = -1 \end{aligned} \tag{4.6}$$

Thus, $y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 0, \forall i$. By rescaling the boundaries, it can be written as

$$\mathsf{y}_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1, \forall i. \tag{4.7}$$

The shortest distance between the two boundaries is $\frac{2}{\|\mathbf{w}\|}$. Thus, the classification problem becomes

$$\begin{aligned} &\min. \ \frac{\|\mathbf{w}\|^2}{2} \\ &\text{subject to} \quad \mathsf{y}_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 \end{aligned} \tag{4.8}$$

which is an optimization problem with a convex quadratic objective and linear constraint functions, and its solution is the optimal margin classifier. To solve this convex optimization problem, Lagrangian approach is set up as

$$L(\mathbf{w}, b, \alpha) = \frac{\|\mathbf{w}\|^2}{2} - \sum_i \alpha_i \left(\mathsf{y}_i(\mathbf{w}^T\mathbf{x}_i + b) - 1\right) \tag{4.9}$$

where α_i is the Lagrangian multiplier. Taking derivative of $L(\mathbf{W}, b, \alpha)$ with respect to $\mathbf{w}$ and b results in

$$\begin{aligned} \mathbf{w} = &\sum_i \alpha_i \mathsf{y}_i \mathbf{x}_i, \\ &\sum_i \alpha_i \mathsf{y}_i = 0 \end{aligned} \tag{4.10}$$

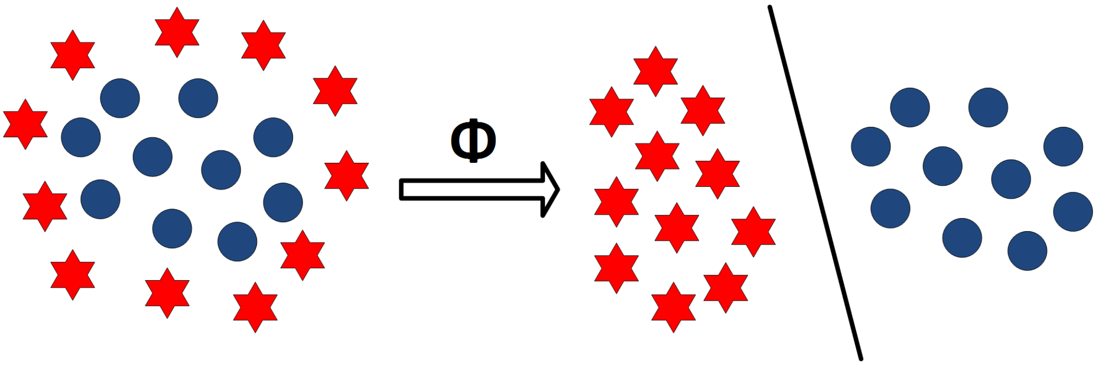

Figure 4.8. An illustration of the kernel trick in the SVM classification.

The dual problem can therefore be written as

$$\begin{aligned} &\max. \ \sum_i \alpha_i - \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j \mathsf{y}_i\mathsf{y}_j \mathbf{x}_i^T\mathbf{x}_j \\ &\text{subject to} \quad \alpha_i \geq 0 \ \text{ and } \ \sum_i \alpha_i \mathsf{y}_i = 0 \end{aligned} \tag{4.11}$$

which is also a quadratic programming problem and can be solved by the QP solver. Note that, the corresponding computer simulations in this research were carried out by LIBSVM (library for SVM) [84]. Once the optimal Lagrange multipliers α_i are found, parameters $\mathbf{w}$ and b can be calculated and the optimal hyperplane is constructed, and the classifier function (predictor) is therefore

$$\begin{aligned} f(\mathbf{x}) &= \text{sign}(\mathbf{w}^T\mathbf{x} + b) \\ &= \text{sign}\left(\sum_i \alpha_i \mathsf{y}_i \mathbf{x}_i^T\mathbf{x} + b\right) \end{aligned} \tag{4.12}$$

Note that The discussion above focuses on the cases, where a perfect linear classification was feasible. However there are occasions, where error-free linear separation is not possible, and thus classification error is inevitable. To this end, by introducing a slack variable $\epsilon_i \geq 0$ to control the intensity of misclassification, the quadratic optimization becomes

$$\begin{aligned}&\min.\ \frac{1}{2}\mathbf{w}^T\mathbf{w}+C\sum_i\epsilon_i\\&\text{subject to}\quad \mathsf{y}_i(\mathbf{w}^T\mathbf{x}_i+b)\geq 1-\epsilon_i,\end{aligned}\tag{4.13}$$

where C is the cost regularization constant. The corresponding dual problem is given as

$$\begin{aligned}&\max.\ \sum_i\alpha_i-\frac{1}{2}\sum_{i,j}\alpha_j\alpha_i\mathsf{y}_i\mathsf{y}_j\mathbf{x}_i^T\mathbf{x}_j\\&\text{subject to}\quad 0\leq\alpha_i\leq C\ \text{ and }\ \sum_i\alpha_i\mathsf{y}_i=0\end{aligned}\tag{4.14}$$

The strength of the SVM is its ability to also classify the non-separable (not linearly separable) data. To this end, SVM maps the input data into another space, in which the data points may become linearly separable, to successfully classify non-separable data. In other words, the original training data points in $\mathcal{R}^d$ are mapped to a new feature space $\mathcal{H}$ via the mapping Φ, in which the maximum margin hyperplane separating the transformed data is computed [76,85]. This concept is illustrated in Figure 4.8. Note that in Equation (4.11), the input data points are utilized in the inner product form. Therefore, if the inner product of the mapped data can be computed in the feature space $\mathcal{H}$, the mapping is not explicitly required. The kernel function is defined as $k(\mathbf{x}_i,\mathbf{x}_j)=\Phi(\mathbf{x}_i)^T\Phi(\mathbf{x}_j)$. It is observed that the kernel in Equation (4.11) is the linear kernel exclusively computed as the inner product of input data. In non-linear scenarios, SVM solves the following quadratic optimization problem

$$\begin{aligned}&\min.\ \frac{1}{2}\mathbf{w}^T\mathbf{w}+C\sum_{i=1}\epsilon_i\\&\text{subject to}\quad \mathsf{y}_i\left(\mathbf{w}^T\Phi(\mathbf{x}_i)+b\right)\geq 1-\epsilon_i\end{aligned}\tag{4.15}$$

where the corresponding dual problem is

$$\begin{aligned}
&\max. \ \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j \mathrm{y}_i \mathrm{y}_j \overbrace{\Phi(\mathbf{x}_i)^T \Phi(\mathbf{x}_j)}^{k(\mathbf{x}_i,\mathbf{x}_j)} \\
&\text{subject to} \quad 0 \leq \alpha_i \leq C \ \text{ and } \sum_i \alpha_i \mathrm{y}_i = 0
\end{aligned} \tag{4.16}$$

The prediction function of the class label is as the following function

$$\begin{aligned}
f(\mathbf{x}) &= \operatorname{sign}\left(\mathbf{w}^T \Phi(\mathbf{x}) + b\right) \\
&= \operatorname{sign}\left(\sum_i \alpha_i \mathrm{y}_i \overbrace{\Phi(\mathbf{x}_i)^T \Phi(\mathbf{x})}^{k(\mathbf{x}_i,\mathbf{x})} + b\right)
\end{aligned} \tag{4.17}$$

There are several non-linear kernel functions, each of which trade-offs different levels of the classification performance and complexity, although in this work, we choose to work with the popular linear kernel and also Gaussian radial basis function (RBF) kernel defined as $k(\mathbf{x_i}, \mathbf{x_j}) = \exp(-\gamma \|\mathbf{x_i} - \mathbf{x_j}\|^2)$ [85].

As mentioned, SVM is inherently a binary classification algorithm, however there are several methods available to extend the idea to multiclass classifiers. We utilize the one-versus-one (OVO) multiclass SVM, in which one binary SVM for each pair of classes is constructed. Therefore, by assuming that there are C_l different classes, $\frac{C_l(C_l-1)}{2}$ binary classifiers are initially trained to distinguish the samples of one class from the samples of another class. Classification decision is finalized according to the majority voting [86].

Training complexity is the major drawback of the standard SVM algorithm, that is, it suffers from a cubic order time complexity, in the worst case for the kernel method, and a space complexity of $\mathcal{O}(n^2)$, where n shows the number of training samples [87]. Therefore, the immediate remedy is to utilize smaller training samples. In the next section, corresponding trade-offs are illustrated via computer simulations.

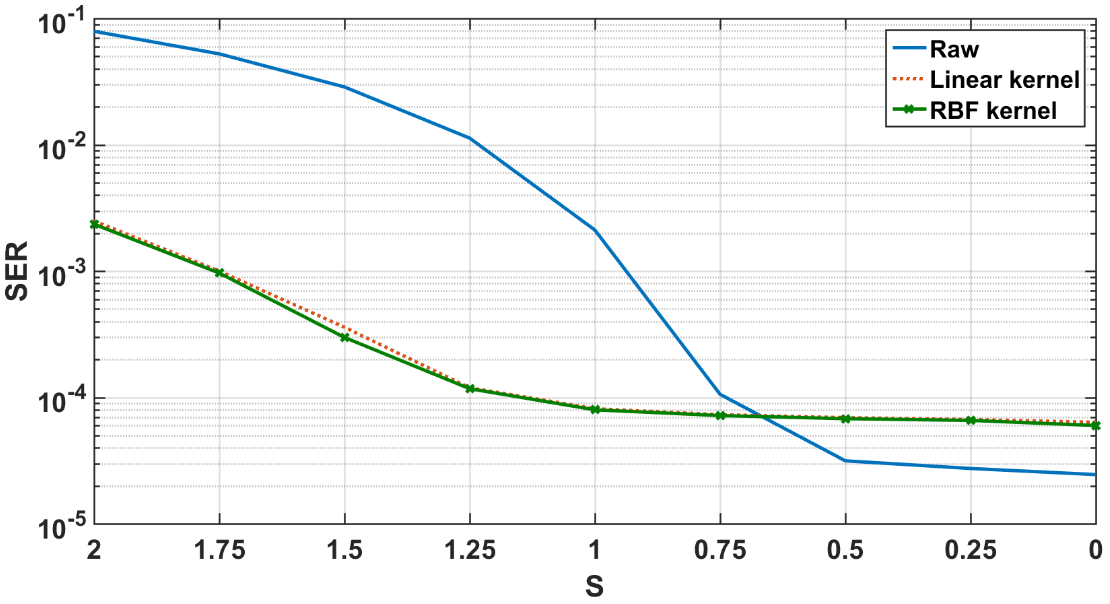

Figure 4.9. SER curves for SVM classification under linear and RBF kernels, where $N_{PE} = 100$ and $T_r = 50$. The training dataset is $\mathbf{x}^1$.

4.2.1. Results and Discussions

In this section, preliminary computer simulation results for the implementation of SVM in the flash memory system are presented, where simulation parameters in Table 3.1 have been employed. In the 2-bit MLC setup, there are 4 different class labels indicating the discrete stored symbols and the multiclash classification is performed via OVO method. Based on the interference structure of the flash channel, the first training vector is constructed as $\mathbf{x}_j^1 = \{y_{i,j}, y_{i+1,j}, y_{i+1,j+1}, y_{i+1,j-1}\}$, where i and j denotes the corresponding page and string (bitline) index of the training observation, respectively. To maintain a minimal computational complexity, the size of the training data is limited to only one page of the block and without loss of generality, channel output observations (sensed threshold voltages) in the first page are employed as the training data, i.e. $i = 1$. Note that, $\mathsf{y}_j = \{s_{1,j}\}$, as the corresponding class labels of the observation in the first page, are also provided to the SVM. To further investigate the impact of different features, the second training data set is chosen as $\mathbf{x}_j^2 = \{y_{1,j}, y_{2,j}, y_{2,j+1}, y_{2,j-1}, y_{3,j}\}$. Moreover, $\mathbf{x}_j^3 = \{y_{1,j}, y_{2,j}, y_{2,j+1}, y_{2,j-1}, y_{3,j}, y_{4,j}\}$, $\mathbf{x}_j^4 = \{y_{1,j}, y_{1,j-1}, y_{1,j+1}, y_{2,j}, y_{2,j+1}, y_{2,j-1}\}$, and $\mathbf{x}_j^5 = \{y_{1,j}\}$. Note that, the last training

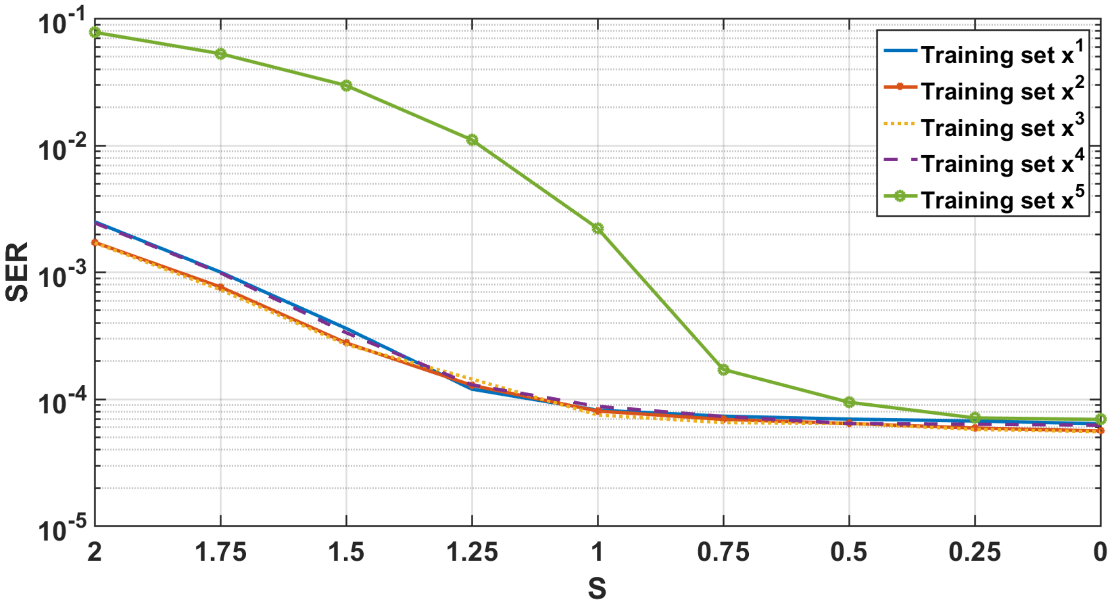

Figure 4.10. SER curves for linear SVM classifications under different training sets, where $N_{PE} = 100$ and $T_r = 50$.

set includes only the cell output without its interferers, which can support the importance of properly training the machine based on the underlying channel structure.

Figure 4.9 shows the simulation results for the SVM classification, where the first training set has been employed. As can bee seen, SVM considerably improves the detection performance compare to the symbol detection, in which ML detection thresholds have been utilized. However, when the interference strength is low, classification performance of the SVM does not further improve, which might be due to the fact that training observations become more uncorrelated, that is independent input samples to the flash channel remain unrelated through the weak interference, and therefore the SVM model exclusively built on the observations' relation thorough interference starts to degrade. Another important remark is that the performance of the RBF SVM negligibly outperforms the linear kernel SVM in the simulated flash channel, which ameliorates the system's complexity requirement by employing the linear kernel. Thus, the rest of experiments have utilized the liner kernel SVM.

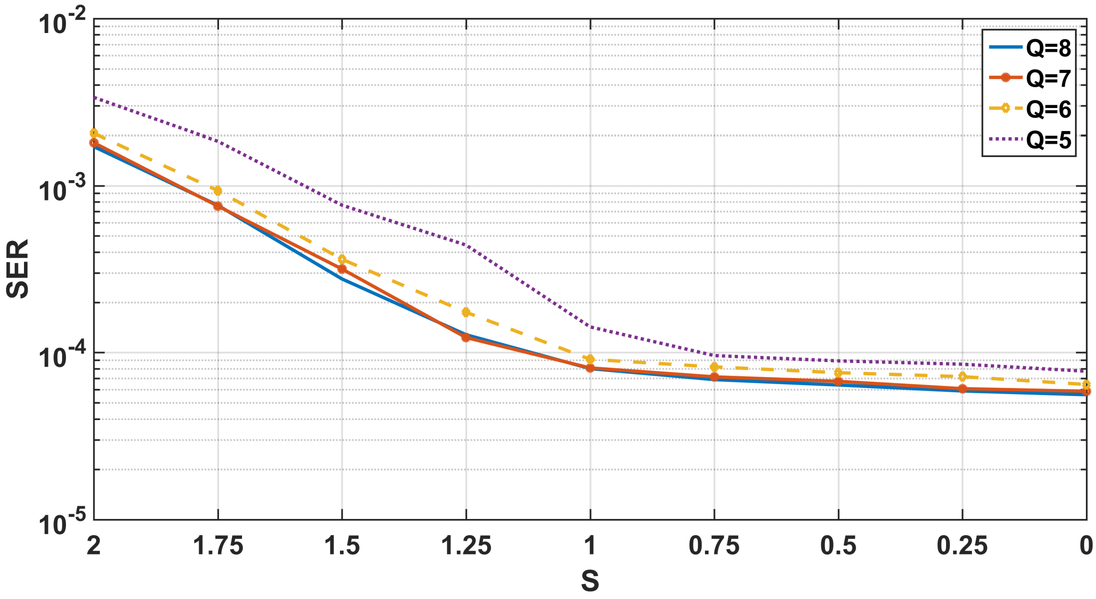

Figure 4.11. SER curves for linear SVM classifications under different sensing resolutions, where $N_{PE} = 100$ and $T_r = 50$. The training dataset is $\mathbf{x}^2$.

Figure 4.10 shows the SER curve comparisons of different training data selections. As can be seen, considering the ICI model for the training set largely improves the classification performance compared to the case, when only the observation (without its interferers) is fed for training the classifier. On the other hand, considering a proper application of prior knowledge of the channel model to the training set construction, different training sets can render negligibly different classification performance. Similar to the experiment in the previous section, Figure 4.11 presents the classification of SVM under different sensing resolution. It is clear that higher sensing resolution results in a better classification, however the system's latency requirement has also to be taken into consideration, as finer sensing resolutions result in higher reading delays, due to the multiple reading attempts.

Employing SVM classification in the flash memory storage systems enables them to adaptively learn and adjust essential parameters according to the latest state of the storage channel, as it is directly trained with observation data, which results in a better error performance. However, important implementation aspects, including computational complexity and read latency have to be considered.

5. CONCLUSION

This dissertation has focused on NAND flash memories, which are currently the ubiquitous and dominant NVM elements. Their main utility is to produce very fast SSDs, which are completely remodeling the disk drives for computers. With recent advances in the area of solid state fabrication, smaller and denser MLC flash memories, with decreasing manufacturing costs, are produced. However, these developments come at the price of lower data reliability and shorter lifespan of the device.

Three main error sources crucially affect the reliability of the MLC NAND flash memory, which are inter-cell interference, retention error, and random telegraph noise. Incorporating their individual effect in a realistic setup to derive a probability distribution model of the erroneously sensed threshold voltage of the memory cell is an important research problem, which directly leads to better detection and error correction performance, thus improving the reliability of the stored data. In this work, first a simplified model of the 4-level cell flash memory, which mainly concentrates on the ICI, was established and the probability distribution of the ICI free and ICI affected threshold voltages were derived. Furthermore, symbol error rate of the aforementioned flash channel was computed in a closed-form expression, which has been corroborated via extensive computer simulations. Furthermore, a very efficient hard-detection based equalization method is proposed to mitigate the ICI problem in the flash channel. To move beyond the scope of simple hard-detection algorithms, a probabilistic setup for the 2D sequence-detection based equalization technique was constructed, which considerably improves the error performance of the storage system. With some modifications, a low-latency and low-complexity 2D sequence detection equalizer has also been proposed.

To build a more accurate probabilistic model for the general MLC flash channels, in which the number of cell levels and interferers can arbitrarily be chosen, erroneous effects of retention and random telegraph noises are combined with the ICI, and conditional PDFs for the sensed threshold voltages were derived in closed-form expressions.

Consequently, error rates of the flash channel have also been computed, which can immediately be utilized in the implementation of superior soft-information based error correcting coding schemes.

A major issue, which has not been previously addressed in the literature, is the strong non-stationarity of error sources in the flash channel, which can largely deviate through time, number of PE cycle, and operation temperature. Thus, relying solely on the derived expressions for the probability distributions and error rate estimations, renders an increasing inaccuracy in the system's performance. To this end and to cope with the non-stationary channel, novel machine learning based intelligent approaches, based on both unsupervised and supervised learning techniques, have been proposed, which periodically learn the underlying probabilistic model of the channel and extract necessary prior information to be utilized in the detection phase. Preliminary computer simulation results have shown that periodic machine learning based algorithms can substantially reduce the error rate, and therefore extend the longevity of the flash device. Furthermore, since they are trained directly from the observations, they can constantly and periodically track the dynamic nature of the channel to adjust the system's performance according to the most recent parameters.

REFERENCES

1. Asadi, M., X. Huang, A. Kavcic and N. P. Santhanam, “Optimal Detector for Multilevel NAND flash Memory Channels with Intercell Interference”, *IEEE Journal on Selected Areas in Communications*, Vol. 32, No. 5, pp. 825–835, May 2014.

2. Cave, A., *What Will We Do When The World’s Data Hits 163 Zettabytes in 2025?*, https://www.forbes.com/sites/andrewcave/2017/04/13/what-will-we-do-when-the-worlds-data-hits-163-zettabytes-in-2025/168ffd34349a, accessed at June 2018.

3. Masuoka, F., M. Asano, H. Iwahashi, T. Komuro and S. Tanaka, “A New Flash E2PROM Cell Using Triple Polysilicon Technology”, *Proceedings of IEEE International Electron Devices Meeting*, Vol. 30, pp. 464–467, December 1984.

4. Sakui, K., “Professor Fujio Masuoka’s Passion and Patience Toward Flash Memory”, *IEEE Solid-State Circuits Magazine*, Vol. 5, No. 4, pp. 30–33, November 2013.

5. Statista, *Size of flash memory market worldwide from 2013 to 2021*, https://www.statista.com/statistics/553556/worldwide-flash-memory-market-size, accessed at June 2018.

6. Chen, S. L., B. R. Ke, J. N. Chen and C. T. Huang, “Reliability Analysis and Improvement for Multi-Level Non-Volatile Memories with Soft Information”, *Proceedings of ACM/EDAC/IEEE Design Automation Conference*, pp. 753–758, June 2011.

7. Cai, Y., E. F. Haratsch, O. Mutlu and K. Mai, “Threshold Voltage Distribution in MLC NAND Flash Memory: Characterization, Analysis, and Modeling”, *Proceedings of the Conference on Design, Automation and Test in Europe*, pp. 1285–1290, March 2013.

8. Lee, D. H. and W. Sung, “Decision Directed Estimation of Threshold Voltage Distribution in NAND flash Memory”, *IEEE Transactions on Signal Processing*, Vol. 62, No. 4, pp. 919–927, February 2014.

9. Dong, G., N. Xie and T. Zhang, “On the Use of Soft-Decision Error-Correction Codes in NAND Flash Memory”, *IEEE Transactions on Circuits and Systems I: Regular Papers*, Vol. 58, No. 2, pp. 429–439, February 2011.

10. Lee, D. H. and W. Sung, “Estimation of NAND Flash Memory Threshold Voltage Distribution for Optimum Soft-Decision Error Correction”, *IEEE Transactions on Signal Processing*, Vol. 61, No. 2, pp. 440–449, January 2013.

11. Li, Q., A. Jiang and E. F. Haratsch, “Noise Modeling and Capacity Analysis for NAND Flash Memories”, *Proceedings of IEEE International Symposium on Information Theory*, pp. 2262–2266, June 2014.

12. Bez, R., E. Camerlenghi, A. Modelli and A. Visconti, “Introduction to Flash Memory”, *Proceedings of the IEEE*, Vol. 91, No. 4, pp. 489–502, April 2003.

13. Kanda, K., N. Shibata, T. Hisada *et al.*, “A 19 nm 112.8 mm^2 64 Gb Multi-Level Flash Memory With 400 Mbit/sec/pin 1.8 V Toggle Mode Interface”, *IEEE Journal of Solid-State Circuits*, Vol. 48, No. 1, pp. 159–167, January 2013.

14. Aritome, S., “NAND Flash Memory Revolution”, *Proceedings of International Memory Workshop*, Vol. 8, pp. 1–4, May 2016.

15. Dirik, C., *Performance Analysis of NAND Flash Memory Solid-State Disks*, Ph.D. Thesis, University of Maryland, College Park, MD, July 2009.

16. Pavan, P., R. Bez, P. Olivo and E. Zanoni, “Flash Memory Cells: An Overview”, *Proceedings of the IEEE*, Vol. 85, No. 8, pp. 1248–1271, August 1997.

17. Dong, G., S. Li and T. Zhang, “Using Data Postcompensation and Predistortion

to Tolerate Cell-to-Cell Interference in MLC NAND Flash Memory", *IEEE Transaction on Circuits and Systems I, Regular Papers*, Vol. 57, No. 10, pp. 2718–2728, October 2010.

18. Suh, K. D., B. H. Suh, Y. H. Lim *et al.*, "A 3.3 V 32 Mb NAND Flash Memory with Incremental Step Pulse Programming Scheme", *IEEE Journal of Solid-State Circuits*, Vol. 30, No. 11, pp. 1149–1156, November 1995.

19. Grupp, L. M., A. M. Caulfield, J. Coburn, S. Swanson, E. Yaakobi, P. H. Siegel and J. K. Wolf, "Characterizing Flash Memory: Anomalies, Observations, and Applications", *Proceedings of IEEE/ACM International Symposium on Microarchitecture*, pp. 24–33, December 2009.

20. Lee, J.-D., S.-H. Hur and J.-D. Choi, "Effects of Floating-Gate Interference on NAND Flash Memory Cell Operation", *IEEE Electron Device Letter*, Vol. 23, No. 5, pp. 264–266, May 2002.

21. Ashrafi, R. A., A. E. Pusane and I. Demirkan, "Analysis of Advanced Programming Architectures for Next-Generation Flash Memories", *Proceedings of Signal Processing and Communications Applications Conference*, pp. 644–644, May 2015.

22. Prall, K., "Scaling Non-Volatile Memory Below 30nm", *Proceedings of IEEE Non-Volatile Semiconductor Memory Workshop*, pp. 5–10, August 2007.

23. Lee, S., T. Kim, K. Kim and J. Kim, "Lifetime Management of Flash-based SSDs Using Recovery-aware Dynamic Throttling", *Proceedings of the 10th USENIX Conference on File and Storage Technologies*, pp. 26–40, February 2012.

24. Grupp, L. M., A. M. Caulfield, J. Coburn, S. Swanson, E. Yaakobi, P. H. Siegel and J. K. Wolf, "Characterizing Flash Memory: Anomalies, Observations, and Applications", *Proceedings of Annual IEEE/ACM International Symposium on Microarchitecture*, pp. 24–33, December 2009.

25. Cai, Y., O. Mutlu, E. F. Haratsch and K. Mai, “Program Interference in MLC NAND Flash Memory: Characterization, Modeling, and Mitigation”, *Proceedings of IEEE International Conference on Computer Design*, pp. 123–130, October 2013.

26. Kim, K., “Future Memory Technology: Challenges and Opportunities”, *Proceedings of International Symposium on VLSI Technology, Systems and Applications*, pp. 5–9, April 2008.

27. Liu, H., S. Groothuis, C. Mouli, J. Li, K. Parat and T. Krishnamohan, “3D Simulation Study of Cell-Cell Interference in Advanced NAND Flash Memory”, *Proceedings of IEEE Workshop on Microelectronics and Electron Devices*, pp. 1–3, April 2009.

28. Cernea, R. A., L. Pham, F. Moogat *et al.*, “A 34 MB/s MLC Write Throughput 16 Gb NAND With All Bit Line Architecture on 56 nm Technology”, *IEEE Journal of Solid-State Circuits*, Vol. 44, No. 1, pp. 186–194, January 2009.

29. Mielke, N., H. Belgal, I. Kalastirsky *et al.*, “Flash EEPROM Threshold Instabilities Due to Charge Trapping During Program/Erase Cycling”, *IEEE Transactions on Device and Materials Reliability*, Vol. 4, No. 3, pp. 335–344, September 2004.

30. Miccoli, C., C. M. Compagnoni, S. Beltrami, A. S. Spinelli and A. Visconti, “Threshold-Voltage Instability Due to Damage Recovery in Nanoscale NAND Flash Memories”, *IEEE Transactions on Electron Devices*, Vol. 58, No. 8, pp. 2406–2414, August 2011.

31. Lee, K., M. Kang, S. Seo, D. H. Li, J. Kim and H. Shin, “Analysis of Failure Mechanisms and Extraction of Activation Energies (E_a) in 21-nm NAND Flash Cells”, *IEEE Electron Device Letters*, Vol. 34, No. 1, pp. 48–50, January 2013.

32. Fukuda, K., Y. Shimizu, K. Amemiya, M. Kamoshida and C. Hu, “Random Telegraph Noise in Flash Memories - Model and Technology Scaling”, *Proceedings of*

IEEE International Electron Devices Meeting, pp. 169–172, December 2007.

33. Tsai, M.-H., T. P. Ma and T. B. Hook, "Channel Length Dependence of Random Telegraph Signal in Sub-Micron MOSFET's", *IEEE Electron Device Letters*, Vol. 15, No. 12, pp. 504–506, December 1994.

34. Kurata, H., K. Otsuga, A. Kotabe *et al.*, "Random Telegraph Signal in Flash Memory: Its Impact on Scaling of Multilevel Flash Memory Beyond the 90-nm Node", *IEEE Journal of Solid-State Circuits*, Vol. 42, No. 6, pp. 1362–1369, June 2007.

35. Fukuda, K., Y. Shimizu, K. Amemiya, M. Kamoshida and C. Hu, "Random Telegraph Noise in Flash Memories - Model and Technology Scaling", *Proceedings of IEEE International Electron Devices Meeting*, pp. 169–172, December 2007.

36. Compagnoni, C. M., M. Ghidotti, A. L. Lacaita, A. S. Spinelli and A. Visconti, "Random Telegraph Noise Effect on the Programmed Threshold-Voltage Distribution of Flash Memories", *IEEE Electron Device Letters*, Vol. 30, No. 9, pp. 984–986, September 2009.

37. Micheloni, R., A. Marelli and K. Eshghi, *Inside Solid State Drives (SSDs)*, Springer, 2012.

38. Withers, C. S., S. Nadarajah and S. H. Shih, "Moments and Cumulants of a Mixture", *Methodology and Computing in Applied Probability*, Vol. 17, No. 3, pp. 541–564, September 2015.

39. Papoulis, A. and S. U. Pillai, *Probability, Random Variables, and Stochastic Processes*, McGraw-Hill Higher Education, 4 edn., 2002.

40. Epstein, B., "Some Applications of the Mellin Transform in Statistics", *Annals of Mathematical Statistics*, Vol. 19, No. 3, pp. 370–379, April 1948.

41. Park, K. T., M. Kang, D. Kim *et al.*, “A Zeroing Cell-to-Cell Interference Page Architecture With Temporary LSB Storing and Parallel MSB Program Scheme for MLC NAND Flash Memories”, *IEEE Journal of Solid-State Circuits*, Vol. 43, No. 4, pp. 919–928, April 2008.

42. Park, D. and J. Lee, “Floating-Gate Coupling Canceller for Multi-Level Cell NAND Flash”, *IEEE Transactions on Magnetics*, Vol. 47, No. 3, pp. 624–628, March 2011.

43. Lee, D. H., J. Kim and W. Sung, “Signal Processing Techniques for Reliability Improvement of Sub-20nm NAND Flash Memory”, *Proceedings of IEEE Workshop on Signal Processing Systems*, pp. 318–323, October 2013.

44. Lee, D. H. and S. Sung, “Least Squares Based Coupling Cancellation for MLC NAND Flash Memory with a Small Number of Voltage Sensing Operations”, *Journal of Signal Processing Systems*, Vol. 71, No. 3, pp. 189–200, December 2013.

45. Kumar, P. S. and S. Roy, “Two-Dimensional Equalization: Theory and Applications to High Density Magnetic Recording”, *IEEE Transactions on Communications*, Vol. 42, No. 234, pp. 386–395, February 1994.

46. Wu, Y., J. A. O’Sullivan, N. Singla and R. S. Indeck, “Iterative Detection and Decoding for Separable Two-Dimensional Intersymbol Interference”, *IEEE Transactions on Magnetics*, Vol. 39, No. 4, pp. 2115–2120, July 2003.

47. Ordentlich, E. and R. M. Roth, “Two-Dimensional Maximum-Likelihood Sequence Detection Is NP Hard”, *IEEE Transactions on Information Theory*, Vol. 57, No. 12, pp. 7661–7670, December 2011.

48. Ashrafi, R. A. and A. E. Pusane, “An Efficient Low-Complexity Two-Dimensional Equalisation Technique for Multi-Level Cell Flash Memory Storage Systems”, *IET Communications*, in press.

49. Forney, G. D., “The Viterbi Algorithm”, *Proceedings of IEEE*, Vol. 61, No. 3, pp.

268–278, March 1973.

50. Rossi, D. and C. Metra, “Error Correcting Strategy for High Speed and High Density Reliable Flash Memories”, *Journal of Electronic Testing*, Vol. 19, No. 5, pp. 511–521, October 2003.

51. Tanzawa, T., T. Tanaka, K. Takeuchi *et al.*, “A Compact On-Chip ECC for Low Cost Flash Memories”, *IEEE Journal of Solid-State Circuits*, Vol. 32, No. 5, pp. 662–669, May 1997.

52. Lin, S. and D. J. Costello, *Error Control Coding, Second Edition*, Prentice-Hall, Inc., Upper Saddle River, USA, 2004.

53. Cai, Y., S. Ghose, E. F. Haratsch, Y. Luo and O. Mutlu, “Error Characterization, Mitigation, and Recovery in Flash-Memory-Based Solid-State Drives”, *Proceedings of the IEEE*, Vol. 105, No. 9, pp. 1666–1704, September 2017.

54. Khayat, P., M. Kaynak, S. Parthasarathy and S. S. Tehrani, “Performance Characterization of LDPC Codes for Large-Volume NAND Flash Data”, *Proceedings of IEEE International Memory Workshop*, pp. 1–4, May 2015.

55. Zigangirov, K. S., A. E. Pusane, D. K. Zigangirov and D. J. Costello, Jr., “On the Error-correcting Capability of LDPC Codes”, *Problems of Information Transmission*, Vol. 44, No. 3, pp. 214–225, September 2008.

56. Hu, X.-Y., M. Fossorier and E. Eleftheriou, “On The Computation of the Minimum Distance of Low-Density Parity-Check Codes”, *Proceedings of IEEE International Conference on Communications*, Vol. 2, pp. 767–771 Vol.2, June 2004.

57. Zhao, K., W. Zhao, H. Sun, T. Zhang, X. Zhang and N. Zheng, “LDPC in SSD: Making Advanced Error Correction Codes Work Effectively in Solid State Drives”, *Proceedings of USENIX Conference on File and Storage Technologies*, pp. 243–256, February 2013.

58. Tanner, R. M., “A Recursive Approach to Low Complexity Codes”, *IEEE Transactions on Information Theory*, Vol. 27, pp. 533–547, June 1981.

59. Miladinovic, N. and M. Fossorier, “Improved Bit-Flipping Decoding of Low-Density Parity-Check Codes”, *IEEE Transactions on Information Theory*, Vol. 51, No. 4, pp. 1594 – 1606, April 2005.

60. Gallager, R. G., *Low-Density Parity-Check Codes*, M.I.T. Press, Cambridge, MA, 1963.

61. Vontobel, P. O. and R. Koetter, “On the Relationship Between Linear Programming Decoding and Min-Sum Algorithm Decoding”, *Proceedings of International Symposium on Information Theory and its Applications*, pp. 991–996, September 2006.

62. Johannesson, R. and K. Sh. Zigangirov, *Fundamentals of Convolutional Coding*, IEEE Press, Piscataway, NJ, 1999.

63. Jiménez-Feltström, A. and K. Sh. Zigangirov, “Time-Varying Periodic Convolutional Codes with Low-Density Parity-Check Matrix”, *IEEE Transactions on Information Theory*, Vol. 45, pp. 2181–2191, September 1999.

64. Lentmaier, M., A. Sridharan, D. J. Costello, Jr. and K. Sh. Zigangirov, “Iterative Decoding Threshold Analysis for LDPC Convolutional Codes”, *IEEE Transactions on Information Theory*, Vol. 56, No. 10, pp. 5274–5289, October 2010.

65. Kudekar, S., C. Méassony, T. Richardsony and R. Urbankez, “Threshold Saturation on BMS Channels via Spatial Coupling”, *International Symposium on Turbo Codes Iterative Information Processing*, pp. 309–313, September 2010.

66. Kudekar, S., T. J. Richardson and R. L. Urbanke, “Threshold Saturation via Spatial Coupling: Why Convolutional LDPC Ensembles Perform So Well Over the BEC”, *IEEE Transactions on Information Theory*, Vol. 57, No. 2, pp. 803–834,

February 2011.

67. Nguyen, P. S., A. Yedla, H. D. Pfister and K. R. Narayanan, "Threshold Saturation of Spatially-Coupled Codes on Intersymbol-Interference Channels", *IEEE International Conference on Communications*, pp. 2181–2186, June 2012.

68. Thorpe, J., "Low-Density Parity-Check (LDPC) Codes Constructed From Protographs", *JPL INP Progress Report*, Vol. 42-154, August 2003.

69. Iyengar, A., M. Papaleo, P. Siegel, J. Wolf, A. Vanelli-Coralli and G. Corazza, "Windowed Decoding of Protograph-Based LDPC Convolutional Codes Over Erasure Channels", *IEEE Transactions on Information Theory*, Vol. 58, No. 4, pp. 2303–2320, April 2012.

70. Ashrafi, R. A. and A. E. Pusane, "Spatially-Coupled Communication System for the Correlated Erasure Channel", *IET Communications*, Vol. 7, No. 8, pp. 755–765, May 2013.

71. Pusane, A. E., R. Smarandache, P. O. Vontobel and D. J. Costello, "Deriving Good LDPC Convolutional Codes from LDPC Block Codes", *IEEE Transactions on Information Theory*, Vol. 57, No. 2, pp. 835–857, February 2011.

72. Costello, D. J., L. Dolecek, T. E. Fuja, J. Kliewer, D. G. M. Mitchell and R. Smarandache, "Spatially Coupled Sparse Codes on Graphs: Theory and Practice", *IEEE Communications Magazine*, Vol. 52, No. 7, pp. 168–176, July 2014.

73. Huang, X., M. Asadi, A. Kavcic and N. P. Santhanam, "All-Bit-Line MLC Flash Memories: Optimal Detection Strategies", *Proceedings of International Conference on Communications*, pp. 3883–3888, June 2014.

74. Cai, Y., E. F. Haratsch, O. Mutlu and K. Mai, "Threshold Voltage Distribution in MLC NAND Flash Memory: Characterization, Analysis, and Modeling", *Proceedings of Design, Automation and Test in Europe Conference*, pp. 1285–1290, March

2013.

75. Dong, G., 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 Systems*, Vol. 20, No. 9, pp. 1705–1714, September 2012.

76. Alpaydin, E., *Introduction to Machine Learning*, The MIT Press, 2014.

77. Mitchell, T. M., *Machine Learning*, McGraw-Hill Inc., New York, USA, 1997.

78. Silverman, B. W., *Density Estimation for Statistics and Data Analysis*, Chapman & Hall, London, 1986.

79. Raykar, V. C., R. Duraiswami and L. H. Zhao, “Fast Computation of Kernel Estimators”, *Journal of Computational and Graphical Statistics*, Vol. 19, No. 1, pp. 205–220, March 2010.

80. Turlach, B. A., “Bandwidth Selection in Kernel Density Estimation: A Review”, *Proceedings of CORE and Institut de Statistique*, February 1999.

81. Jones, M. C., J. S. Marron and S. J. Sheather, “A Brief Survey of Bandwidth Selection for Density Estimation”, *Journal of the American Statistical Association*, Vol. 91, No. 433, pp. 401–407, March 1996.

82. Jia, P., H.-Y. Zhang and X. Z. Shi, “Blind Source Separation Based on Non-parametric Density Estimation”, *Circuits, Systems and Signal Processing*, Vol. 22, No. 1, pp. 57–67, January 2003.

83. Boser, B. E., I. M. Guyon and V. N. Vapnik, “A Training Algorithm for Optimal Margin Classifiers”, *Proceeding of Annual Workshop on Computational Learning Theory*, pp. 144–152, July 1992.

84. Chang, C. C. and C. J. Lin, “LIBSVM: A Library for Support Vector Machines”,

ACM Transactions on Intelligent Systems and Technology, Vol. 2, pp. 27:1–27:27, April 2011.

85. Campbell, C., “Kernel Methods: A Survey of Current Techniques”, *Neurocomputing*, Vol. 48, No. 1, pp. 63 – 84, October 2002.

86. Hsu, C. W. and C. J. Lin, “A Comparison of Methods for Multiclass Support Vector Machines”, *IEEE Transactions on Neural Networks*, Vol. 13, No. 2, pp. 415–425, March 2002.

87. Fine, S. and K. Scheinberg, “Efficient SVM Training Using Low-Rank Kernel Representations”, *Journal of Machine Learning Research*, Vol. 2, pp. 243–264, March 2002.