THE HILBERT SPACE OF PROBABILITY MASS FUNCTIONS AND APPLICATIONS ON PROBABILISTIC INFERENCE

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

MUHAMMET FATİH BAYRAMOĞLU

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF DOCTOR OF PHILOSOPHY
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

SEPTEMBER 2011

Approval of the thesis:

THE HILBERT SPACE OF PROBABILITY MASS FUNCTIONS AND APPLICATIONS ON PROBABILISTIC INFERENCE

Submitted by **MUHAMMET FATİH BAYRAMOĞLU** in partial fulfillment of the requirements for the degree of **Doctor of Philosophy in Electrical and Electronics Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. İsmet Erkmen
Head of Department, **Electrical and Electronics Engineering** _____

Assoc. Prof. Dr. Ali Özgür Yılmaz
Supervisor, **Electrical and Electronics Engineering Dept., METU** _____

Examining Committee Members:

Prof. Dr. Yalçın Tanık
Electrical and Electronics Eng. Dept., METU _____

Assoc. Prof. Dr. Ali Özgür Yılmaz
Electrical and Electronics Eng. Dept., METU _____

Prof. Dr. Mustafa Kuzuoğlu
Electrical and Electronics Eng. Dept., METU _____

Assoc. Prof. Dr. Emre Aktaş
Electrical and Electronics Eng. Dept., Hacettepe University _____

Assist. Prof. Dr. Ayşe Melda Yüksel
Electrical and Electronics Eng. Dept.,
TOBB University of Economy and Technology _____

Date: _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: MUHAMMET FATİH BAYRAMOĞLU

Signature :

ABSTRACT

THE HILBERT SPACE OF PROBABILITY MASS FUNCTIONS
AND APPLICATIONS ON PROBABILISTIC INFERENCE

Bayramoğlu, Muhammet Fatih
Ph.D., Department of Electrical and Electronics Engineering
Supervisor : Assoc. Prof. Dr. Ali Özgür Yılmaz

September 2011, 123 pages

The Hilbert space of probability mass functions (pmf) is introduced in this thesis. A factorization method for multivariate pmfs is proposed by using the tools provided by the Hilbert space of pmfs. The resulting factorization is special for two reasons. First, it reveals the algebraic relations between the involved random variables. Second, it determines the conditional independence relations between the random variables. Due to the first property of the resulting factorization, it can be shown that channel decoders can be employed in the solution of probabilistic inference problems other than decoding. This approach might lead to new probabilistic inference algorithms and new hardware options for the implementation of these algorithms. An example of new inference algorithms inspired by the idea of using channel decoder for other inference tasks is a multiple-input multiple-output (MIMO) detection algorithm which has a complexity of the square-root of the optimum MIMO detection algorithm.

Keywords: The Hilbert space of pmfs, factorization of pmfs, probabilistic inference, MIMO detection, Markov random fields

ÖZ

OLASILIK KÜTLESİ FONKSİYONLARININ HİLBERT UZAYI
VE OLASILISIKSAL BİLGİ ÇIKARIMI ÜZERİNE UYGULAMALARI

Bayramoğlu, Muhammet Fatih
Doktora, Elektrik Elektronik Mühendisliği Bölümü
Tez Yöneticisi : Doç. Dr. Ali Özgür Yılmaz

Eylül 2011, 123 sayfa

Bu tezde olasılık kütlesi fonksiyonlarının Hilbert uzayı sunulmaktadır. Bu Hilbert uzayının sağladığı olanaklar kullanılarak çok değişkenli olasılık kütlesi fonksiyonlarını çarpanlarına ayırmak için bir yöntem önerilmiştir. Bu yöntemden elde edilen çarpanlara ayırma iki nedenle özeldir. İlk olarak, bu çarpanlara ayırma rastgele değişkenler arasındaki cebirsel bağıntıları ortaya koyar. İkinci olarak, rastgele değişkenler arasındaki koşullu bağımsızlık ilişkilerini belirler. Birinci özellik sayesinde kanal kod çözücülerinin, kod çözmekten başka olasılıksal bilgi çıkarımı problemlerinin çözümünde de kullanılabileceği gösterilebilir. Bu yaklaşım yeni olasılıksal bilgi çıkarımı algoritmalarına ve bu algoritmaları gerçeklemek için yeni donanım olanaklarına yol açabilir. Kod çözücülerin kod çözmekten başka bilgi çıkarımı görevlerinde kullanılması fikrinden esinlenen algorıtmaların bir örneği, karmaşıklığı en iyi algoritmanın karekökü olan bir çok-girdili çok-çıktılı sezim algoritmasıdır.

Anahtar Kelimeler: Olasılık kütlesi fonksiyonlarının Hilbert uzayı, Olasılık kütlesi fonksiyonlarının çarpanlara ayrılması, olasılısıksal bilgi çıkarımı, çok-girdili çok-çıktılı sezim, Markov rastgele alanlar

Karıma
To my wife

ACKNOWLEDGMENTS

Firstly, I would like to thank sincerely my supervisor Assoc. Prof. Dr. Ali Özgür Yılmaz. He is an exception in this department regarding both his scientific vision and his personality. His trust and encouragement was crucial to me while working on this thesis.

I would like to thank the members of the thesis progress monitoring committee members Prof. Dr. Mustafa Kuzuoğlu and Assoc. Prof. Dr. Emre Aktaş for their valuable comments and contributions. Moreover, operator theory course of Prof. Kuzuoğlu helped me a lot in this thesis. Furthermore, I appreciate the financial support that Assoc. Prof. Aktaş provided to me in the last periods of my thesis study and his understanding. I would like to thank also the rest of thesis jury members Prof. Dr. Yalçın Tanık and Assist. Prof. Dr. Melda Yüksel for their valuable comments.

I would like express my gratitude to the other two exceptional faculty members of this department who are Prof. Dr. Arif Ertaş and Assoc. Prof. Dr. Çağatay Candan. Prof. Ertaş is a person really deserving his name "Arif". Assoc. Prof. Candan is one of the easiest persons that I can communicate with and I appreciate his "always open" door.

I would like to thank Prof. Dr. Zafer Ünver. I learned a lot from him while assisting EE213 and EE214 courses.

Education is a long haul run. Hence, sincere thanks go to my high school mathematics teachers Ahmet Cengiz, Hüseyin Çakır, Perihan Özdingiş, and of course Demir Demirhas. A special thanks goes to my undergraduate advisor Prof. Dr. Gönül Turhan Sayan.

I would like to acknowledge the free software community. I have never needed and used any commercial software during my Ph.D. research.

I would like to thank Dr. Jorge Cham for phdcomics which introduced some smiles to our overly stressed lives.

I would like to thank my friends Alper Söyler, Murat Kılıç, Mehmet Akif Antepli, Murat

Üney, Yılmaz Kalkan, Serdar Gedik, and Onur Özeç for their valuable friendship.

My sincere gratitude goes to my parents Nezahat and Mustafa Bayramoğlu. This thesis could not finish without their prayers. But the good manners I learned from them is much more valuable to me than this Ph.D. degree. I would like to thank my brother Etka for being the kindest brother in the world.

My deepest thanks goes to my wife Neslihan, or more precisely Dr. Neslihan Yalçın Bayramoğlu. I appreciate everything she sacrificed for me. I studied on this thesis on times that I stoled from her and she really deserves at least the half of the credit for this thesis. Her support not only was vital for me during the Ph.D. but also will continue to be vital during the rest of my life.

PREFACE

This thesis summarizes the research work carried out in six years starting from September 2005. The research topic arose while I was trying to develop an analysis method for the convergence rate of the iterative sum-product algorithm. Since the messages (beliefs) passed between the nodes in the iterative sum-product algorithm are probability mass functions (pmf), I thought that representing the pmfs in a Hilbert space structure would prove useful in the analysis of the sum-product algorithm. Analyzing the convergence of the sum-product algorithm would be an application of the norm in the Hilbert space of pmfs. However, later I noticed that the inner product has much more interesting applications and preferred focusing on the applications of the inner product to dealing with the convergence which led to this thesis.

In order to read the thesis a basic understanding of inner product spaces and finite fields is necessary. Anybody with this background can follow the chapters from the second to the fifth. I believe that these chapters are the core of the thesis. Chapter 6 contains some applications from communication theory and might require a communication theory background.

This preface is an adequate place to note some observations about my country and university. I am happy to observe that Turkey improved economically and democratically during my graduate studies. On the other hand, I am sad to observe that Middle East Technical University downgraded scientifically and democratically during the same time.

This thesis is related probability theory. Probability theory is an area which is close to the border between science and belief. Although Laplace's book on celestial mechanics misses to mention God, my explanation on the relation between probability and willpower makes me to believe in God and I would like to start to the rest of the thesis by a quote from the translation of Qur'an which explains what is science to me: "Glory be to You, we have no knowledge except what you have taught us. Verily, it is You (Allah), the All-Knower, the All-Wise".

TABLE OF CONTENTS

LIST OF FIGURES

FIGURES

CHAPTER 1

INTRODUCTION

1.1 Motivation

A linear vector space structure over a set provides algebraic tools such as addition and scaling to carry out on the elements of the set. If a vector space can be endowed with an inner product then it becomes an inner product space. An inner product provides geometric concepts such as norm, distance, angle, and projections. If every Cauchy sequence in an inner product space converges with respect to the inner product induced norm then the inner product space becomes a Hilbert space. Needless to say a Hilbert space structure is very useful and find application areas in diverse fields of science. Communication theory is not an exception. For instance, the signal space representation in communication theory relies on the Hilbert space structure constructed over the set of square integrable functions.

One of the mathematical objects that is too frequently used in communication and information theories is the probability mass functions (pmf) which are discrete equivalents of probability density functions. Although, pmfs are so frequently used in communication and information theories a Hilbert space structure for them was missing. A Hilbert space of pmfs might have many interesting applications.

A possible application for the Hilbert space of probability mass functions might be analyzing the characteristics of a multivariate pmf. An important characteristic of a multivariate pmf is the conditional independence relations imposed by it. The conditional independence relation imposed by a multivariate pmf is determined by the factorization of the pmf to local functions[1] as explained in [18, 19].

[1] Local functions are functions (not necessarily pmfs) which have less arguments than the original multivariate pmf.

The factorization structure of a multivariate pmf into local functions also determines the algorithms which can perform inference on the pmf, in other words, maximize or marginalize the pmf. The sum-product algorithm, which is also called belief propagation, and the max-product algorithm effectively marginalize or maximize a multivariate pmf by exploiting the pmfs' factorization structure [1]. Modern decoding algorithms such as low-density parity-check decoding and turbo decoding, which have become highly popular in the last decade, relies on this fact.

Some multivariate pmfs, for instance the pmf resulting from a hidden Markov model, has an apparent factorization structure. However, one cannot be sure whether this factorization structure is the "best" possible factorization or not. On the other hand, some pmfs, for instance the pmfs obtained empirically, might not have an apparent factorization structure at all. Therefore, developing a method which obtains the factorization of a multivariate pmf systematically would prove useful in many areas.

1.2 Contributions

The first contribution in this thesis is the derivation of the Hilbert space structure for pmfs. The Hilbert space of pmfs not only provides a vectorial representation of evidence but also it proves to be a useful tool in analyzing the pmfs.

The second contribution of this thesis is a systematic method for obtaining factorization of a multivariate pmf. The resulting factorization is unique and is the ultimate factorization possible. Hence, we call the resulting factorization as the canonical factorization. The canonical factorization of a multivariate pmf is obtained by projecting the pmf onto orthogonal basis pmfs of the Hilbert space of pmfs. Hence, this factorization method heavily relies on the Hilbert space of pmfs.

The basis pmfs mentioned in the paragraph above are special pmfs such that their value is determined only by a linear combination of their arguments. In order to be able to talk about linear combinations of arguments addition and multiplication must be well defined between arguments of the pmf. Hence, the canonical factorization of a pmf can be obtained only if the pmf is a pmf of finite-field-valued random variables. This is an important limitation of the canonical factorization.

The property of the basis pmfs mentioned in the previous paragraph causes an important limitation but also this property leads to the third and the probably the most important contribution of the thesis. Since the basis pmfs are functions of their arguments, the canonical factorization reveals the algebraic dependencies between the random variables. Thanks to this fact, it can be shown that channel decoders can be employed as an apparatus for tasks beyond decoding. This idea leads to new hardware options as well as new inference algorithms.

The fourth contribution of the thesis is an application of the idea explained in the paragraph above. This contribution is a multiple-input multiple-ouput (MIMO) detection algorithm which employs the decoder of a tail biting convolutional code as a processing device. This algorithm is an approximate soft-input soft-output MIMO detection algorithm whose complexity is the square-root of that of the optimum MIMO detection algorithm.

The final contribution of the thesis is another property of the canonical factorization. It can be shown that the conditional dependence relationships imposed by a multivariate pmf can be determined from the canonical factorization of the pmf. In other words, the conditional independence relationships imposed by a pmf can be determined by using the geometric tools provided by the Hilbert space of pmfs. This property of the canonical factorization might lead to applications in experimental fields such as bioinformatics dealing with large amounts of data.

1.3 Comparison to earlier work

A Hilbert space of probability density functions is first presented in literature in a very different area of science, stochastic geology, in [4]. Their derivation is for a class of continuous probability density functions. On the other hand our derivation is for pmfs. Although, the resulting Hilbert space structures in both their and our derivations are quite similar, our derivation is independent of theirs. Furthermore, we provide many applications of the Hilbert space of pmfs on probabilistic inference.

The canonical factorization proposed in this thesis can be compared to the factorization of pmfs provided by the Hammersley-Clifford theorem [18, 19]. Both the Hammersley-Clifford theorem and the canonical factorization can completely determine the conditional independence relationships imposed by a pmf. But Hammersley-Clifford theorem does not highlight

the algebraic dependence relationships between random variables while the canonical factorization does. Moreover, the canonical factorization is unique whereas the factorization of the Hammersley-Clifford theorem is not.

The results obtained in this thesis can be located in the factor graph literature as follows. Factor graphs are bipartite graphical models which represent the factorization of a pmf [1]. The bipartite graphs were first employed by Tanner to describe low complexity codes in [5]. A very crucial step in achieving the factor graph representation is the Ph.D. thesis of Wiberg [6, 7]. In his thesis Wiberg showed the connection between various codes and decoding algorithms by introducing hidden state nodes to the graphs described by Tanner and characterized the message passing algorithms running on these graphs. Local constraints in [6] are behavioral constraints, such as parity check constraints. The factor graphs are the generalization of the graphical models introduced in [6] by allowing local constraints to be arbitrary functions rather than behavioral constraints [1].

The canonical factorization proposed in this thesis can also be represented by a factor graph. Moreover, the factor functions appearing in the canonical factorization can be transformed into usual parity check constraints by introducing some auxiliary variables. Therefore, the factor graph representing the canonical factorization can be transformed into a Tanner graph by introducing some auxiliary variable nodes which are very different from the hidden state nodes introduced in [6]. This is essentially an explanation of the claim that the channel decoders can be employed for inference tasks beyond decoding.

1.4 Outline

After this chapter, the thesis continues with the introduction of the Hilbert space of pmfs in Chapter 2. The Hilbert space of pmfs is the main tool to be used throughout the thesis. The canonical factorization is introduced in Chapter 3. Chapter 4 investigates the properties and special cases of the canonical factorization. Chapter 5 explains how a channel decoder can be used for other probabilistic inference tasks other than its own purpose. This explanation is based on the canonical factorization. Some possible consequences of this result are also explained in Chapter 5. Chapter 6 provides some basic examples from communication theory on the use of channel decoders for other inference tasks beyond decoding. The MIMO detec-

tor which uses the decoder of a tail biting convolutional code is also introduced in this chapter. Chapter 7 shows that the conditional independence relations can be completely determined from the canonical factorization. The thesis is concluded with some possible future directions in Chapter 8. For the sake of neatness of the thesis some proofs and derivations are collected in the Appendix.

1.5 Some remarks on notation

Throughout the thesis we denote the deterministic variables with lowercase letters and random variables with uppercase letters. We represent functions of multiple variables as functions of vectors and denote vectors with boldface letters. Lowercase boldface letters denote deterministic vectors and capital boldface letters denote random variables. All vectors encountered in the thesis are row vectors except a few cases in Chapter 6.

Matrices are also denoted with capital boldface letters which might lead to a confusion with random vectors. Throughout the thesis, we used $\mathbf{V}$, $\mathbf{W}$, $\mathbf{X}$, $\mathbf{Y}$, and $\mathbf{Z}$ to denote random vectors. All the other capital boldface letters are matrices.

Unfortunately, many different types of additions are included in the thesis such as finite field addition, real number addition, vector addition, and even direct sum of subspaces. We reserve $\oplus$ symbol for the direct sum of subspaces for the sake of consistency with the linear algebra literature. We use $\boxplus$ symbol for the vectorial addition operation of pmfs which is defined in Chapter 2. We have to use the remaining + symbol for all the rest of addition operations such as real number addition, finite field addition, and vectorial addition in $\mathbb{R}^N$. Fortunately, the type of the addition employed can be determined from the types of the operands.

A possible confusion might arise while using the summation symbol $\sum$. For instance, $\sum_{i=1}^{N} p_i(x)$ might refer to both $p_1(x)+p_2(x)+\ldots+p_N(x)$ and $p_1(x)\boxplus p_2(x)\boxplus\ldots\boxplus p_N(x)$ which are really two different summations. In order to avoid this confusion we denote the latter summation with $\overset{\boxplus}{\sum_{i=1}^{N}} p_i(x)$, although summations like the former is never encountered in the thesis.

CHAPTER 2

THE HILBERT SPACE OF PROBABILITY MASS FUNCTIONS

2.1 Introduction

The Hilbert space of probability mass functions (pmf), which is the main tool to be employed in the thesis, is introduced in this chapter. Throughout the thesis we are only interested in the pmfs of the finite-field-valued random variables. Therefore, we define what a finite-field-valued random variable is first in Section 2.2. We introduce the set of pmfs on which we construct the Hilbert space in Section 2.3. Then we construct the algebraic and geometric structures over this set in Section 2.4 and Section 2.5 respectively. Section 2.6 emphasizes the differences between the Hilbert space of random variables and the Hilbert space of pmfs in order to avoid possible confusion. Finally, in Section 2.7 the idea of the construction of the Hilbert space is repeated on the set of multivariate pmfs.

2.2 Finite-Field-Valued Random Variables

Traditionally a random variable is a mapping from the event space to the real or complex fields. However, in some experiments, e.g., the experiments with discrete event spaces, it might be useful to map the outcomes of the experiment to a finite (Galois) field. Such a mapping would allow to carry out *meaningful* algebraic operations between the outcomes of different experiments, for instance as in [32]. A finite-field-valued random variable is defined below.

Definition 1 Finite-field-valued random variable: *Let* Ω *be the event space of an experiment and* $\mathbb{F}_q = \mathrm{GF}(q)$ *be the finite field of* q *elements. Moreover, let a function* $X : \Omega \to \mathbb{F}_q$ *be*

defined as

$$X(\omega \in \mathcal{E}_i) \triangleq i \quad \forall i \in \mathbb{F}_q,$$

where $\{\mathcal{E}_i : i \in \mathbb{F}_q\}$ *are events (subsets of* Ω*) of this experiment. The function X is called an* $\mathbb{F}_q$*-valued random variable if the events* $\{\mathcal{E}_i : i \in \mathbb{F}_q\}$ *are mutually exclusive and collectively exhaustive, i.e.,*

$$\begin{aligned} \mathcal{E}_i \neq \mathcal{E}_j \quad &\Longrightarrow \quad \mathcal{E}_i \cap \mathcal{E}_j = \emptyset \quad \forall i, j \in \mathbb{F}_q, \\ \bigcup_{i \in \mathbb{F}_q} \mathcal{E}_i \quad &= \quad \Omega. \end{aligned}$$

Actually, we do not need to restrict ourselves to the finite-field-valued random variables in this chapter since the ideas presented in this chapter can be applied to any discrete random variable. We need the concept of finite-field-valued random variables starting from the next chapter. However, we introduce the finite-field-valued random variables starting from this chapter in order to make the representation simpler.

2.3 The Set of Strictly Positive Probability Mass Functions

Many different experiments can be represented with an $\mathbb{F}_q$-valued random variable. All these experiments may lead to different pmfs. Furthermore, we may have different pmfs even for the same experiment if the outcome is conditioned on some other event. Let $\mathcal{P}_{\mathbb{F}_q}$ be the set of all *strictly positive* pmfs that an $\mathbb{F}_q$-valued random variable might possess, i.e.,

$$\mathcal{P}_{\mathbb{F}_q} \triangleq \left\{ p(x) : \mathbb{F}_q \to (0,1) \subset \mathbb{R} \text{ s.t. } \sum_{x \in \mathbb{F}_q} p(x) = 1 \right\}. \tag{2.1}$$

The Hilbert space of pmfs is going to be constructed on $\mathcal{P}_{\mathbb{F}_q}$. This set excludes the pmfs which take value zero for some values. The reason under this restriction will be clear after scalar multiplication is defined on this set.

We are going to represent the pmfs with lowercase letters such as $p(x)$, $r(x)$, or $s(x)$. These pmfs may represent the pmfs of random variables representing different experiments as well as they may represent the pmfs of the same random variable conditioned on different events.

2.3.1 The normalization operator

We employ a normalization operator to obtain pmfs from strictly positive real-valued functions by scaling them. We denote this normalization operator with $C_{\mathbb{F}_q}\{.\}$ and define it as

$$C_{\mathbb{F}_q}\{\alpha(x)\} : \mathcal{F}_{\mathbb{F}_q} \to \mathcal{P}_{\mathbb{F}_q} \triangleq \frac{\alpha(x)}{\sum_{i\in\mathbb{F}_q}\alpha(i)}, \tag{2.2}$$

where the set $\mathcal{F}_{\mathbb{F}_q}$ denotes the set of all functions from $\mathbb{F}_q$ to $\mathbb{R}^+$ and $\alpha(x)$ is a function in $\mathcal{F}_{\mathbb{F}_q}$. An obvious property of the operator $C_{\mathbb{F}_q}\{.\}$ that we exploit frequently is given below

$$C_{\mathbb{F}_q}\{\beta\alpha(x)\} = C_{\mathbb{F}_q}\{\alpha(x)\}, \tag{2.3}$$

where β is any positive number.

2.4 The Algebraic Structure over $\mathcal{P}_{\mathbb{F}_q}$

The foundation of the Hilbert space of PMFs is the addition operation. Hence, the definition of the addition should be meaningful in the sense of probabilistic inference in order to take advantage of the Hilbert space structure for inference problems.

The addition operation is inspired by the following scenario. Assume that we receive information about a uniformly distributed source X via two *independent* channels with outputs y_1 and y_2 as depicted in Figure 2.1. Let $p(x) = \Pr\{X = x|y_1\}$, $q(x) = \Pr\{X = x|y_2\}$, and $r(x) = \Pr\{X = x|y_1, y_2\}$. Since X is uniformly distributed, $r(x)$ can be derived as

$$\begin{aligned} r(x) &= \frac{p(x)q(x)}{\sum_{x\in\mathbb{F}_q} p(x)q(x)} && (2.4)\\ &= C_{\mathbb{F}_q}\{p(x)q(x)\} && (2.5) \end{aligned}$$

by employing the Bayes' theorem.

The PMFs $p(x)$ and $q(x)$ represent the evidence about the source X when only y_1 or y_2 is known respectively. On the other hand, $r(x)$ represents the total evidence when both outputs are known. In a way, $r(x)$ is obtained by summing $p(x)$ and $q(x)$. Hence, (2.4) can be adopted as the definition of addition. For any $p(x)$ and $q(x)$ in $\mathcal{P}_{\mathbb{F}_q}$ their addition is denoted by $\boxplus$ and defined as

$$p(x) \boxplus q(x) \triangleq C_{\mathbb{F}_q}\{p(x)q(x)\}. \tag{2.6}$$

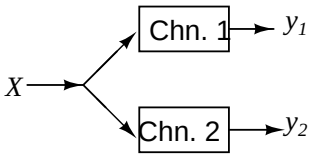

Figure 2.1: The scenario for explaining the meaning of addition operation.

The definition of the addition operation is such a critical point of this thesis that the rest of the thesis will be built upon this definition.

This definition of addition operation is the same as parallel information combining operation as defined in [13] and message computation at variable nodes in the sum-product algorithm [1].

Defining the addition operation also enforces the scalar multiplication to have such a form that scalar multiplication is consistent with the addition. The scalar multiplication, which is denoted by $\boxtimes$, should satisfy the relation below for positive integers n

$$\begin{aligned} n \boxtimes p(x) &= \underbrace{p(x) \boxplus p(x) \boxplus \ldots \boxplus p(x)}_{\text{n times}} \\ &= C_{\mathbb{F}_q} \{(p(x))^n\} . \end{aligned} \tag{2.7}$$

Generalizing (2.7) to any α in $\mathbb{R}$ leads to the definition of scalar multiplication below

$$\alpha \boxtimes p(x) \triangleq C_{\mathbb{F}_q} \{(p(x))^\alpha\} . \tag{2.8}$$

In order to be able to scale $p(x)$ with negative coefficients it is necessary that $p(x) \neq 0$ for any x in $\mathbb{F}_q$. Hence, in the definition of $\mathcal{P}_{\mathbb{F}_q}$ an open interval is used rather than a closed interval in (2.1).

Theorem 2.1 *The set $\mathcal{P}_{\mathbb{F}_q}$ together with operations $\boxplus$ and $\boxtimes$ forms a linear vector space over $\mathbb{R}$.*

Proof. The closure of $\mathcal{P}_{\mathbb{F}_q}$ under both operations is ensured by the normalization operators in their definitions. The commutativity and associativity are obvious from the definition of $\boxplus$ operation. The neutral element with respect to (w.r.t.) the addition operation is the uniform

distribution given by

$$\theta(x) = \frac{1}{q}.$$

Consequently, the additive inverse of $p(x)$, which is denoted by $\boxminus p(x)$, is

$$\boxminus p(x) = C_{\mathbb{F}_q}\left\{\frac{1}{p(x)}\right\} = -1 \boxtimes p(x).$$

The compatibility of scalar multiplication with the multiplication in $\mathbb{R}$ is obvious from (2.8). The distributivity of multiplication over scalar and vector additions are direct consequences of the definitions of scalar multiplication and addition. Clearly, 1 is the identity element of scalar multiplication. Hence, $\mathcal{P}_{\mathbb{F}_q}$ becomes a *linear vector space* over $\mathbb{R}$. ■

Example 2.1 *The algebraic relations between some conditional pmfs is examined in this example in which a combined experiment is taking place in a two dimensional universe.*

First a fair die with three faces[1] *is rolled. Then one of the three urns is selected corresponding to the outcome of the die rolling experiment. These three urns contain balls of six different colors. The number of balls of different colors in each urn is given in the table below. A ball is drawn from the selected urn and replaced back a few times.*

Table 2.1: Number of balls in different colors in each urn mentioned in Example 2.1.

	Red (R)	Yellow (Y)	Orange (O)	Blue (B)	Green (G)	Purple (P)
Urn 1	1	9	9	3	1	1
Urn 2	9	1	9	1	3	1
Urn 3	9	9	1	1	1	3

Let the event space of the die rolling experiment be mapped to a $\mathbb{F}_3$-valued random variable X such that the faces 1, 2, *and* 3 *are mapped to* 0, 1, *and* 2 *in $\mathbb{F}_3$. Let six pmfs of X conditioned on the color of the ball drawn be defined as follows when a single ball is drawn.*

$$\begin{aligned} r(x) &\triangleq \Pr\{X = x|\, R\,\} \quad y(x) \triangleq \Pr\{X = x|\, Y\,\} \quad o(x) \triangleq \Pr\{X = x|\, O\,\} \\ b(x) &\triangleq \Pr\{X = x|\, B\,\} \quad g(x) \triangleq \Pr\{X = x|\, G\,\} \quad p(x) \triangleq \Pr\{X = x|\, P\,\} \end{aligned} \tag{2.9}$$

For instance, assume that a ball is drawn from the selected urn and replaced back six times and the colors of the balls drawn are B, B, G, G, G, and Y. Then the a posteriori pmf of X can

[1] We can have a die with three faces in a two dimensional universe. This is the reason why the experiment takes place in a two dimensional universe.

be expressed by using the definitions of addition and scalar multiplication in $\mathcal{P}_{\mathbb{F}_3}$ *as*

$$\Pr\{X = x|B, B, G, G, G, Y\} = 2 \boxtimes b(x) \boxplus 3 \boxtimes g(x) \boxplus y(x).$$

Now assume that the process of drawing a ball and replacing is repeated three times and the colors of the drawn balls are R, Y, and O. Then due to the symmetry in the problem the a posteriori pmf of X is

$$\Pr\{X = x|R, Y, O\} = \frac{1}{3}.$$

Vectorial representation of this equation in $\mathcal{P}_{\mathbb{F}_3}$ *is*

$$r(x) \boxplus y(x) \boxplus o(x) = \theta(x). \tag{2.10}$$

Similarly, $b(x)$*,* $g(x)$*, and* $p(x)$ *are also related as*

$$b(x) \boxplus g(x) \boxplus p(x) = \theta(x). \tag{2.11}$$

Now assume that the process of drawing a ball and replacing is repeated twice. The a posteriori pmf of X given the colors of the balls are R and Y is

$$\Pr\{X = x|R, Y\} = r(x) \boxplus y(x) = \begin{cases} 1/11 & , \quad x = 0 \\ 1/11 & , \quad x = 1 \\ 9/11 & , \quad x = 2 \end{cases} \tag{2.12}$$

and the a posteriori pmf of X given both balls are P is

$$\Pr\{X = x|P, P\} = 2 \boxtimes p(x) = \begin{cases} 1/11 & , \quad x = 0 \\ 1/11 & , \quad x = 1 \\ 9/11 & , \quad x = 2 \end{cases}. \tag{2.13}$$

Combining these last two results yields

$$r(x) \boxplus y(x) = 2 \boxtimes p(x). \tag{2.14}$$

The following two relations can be obtained similarly.

$$r(x) \boxplus o(x) = 2 \boxtimes g(x) \tag{2.15}$$

$$o(x) \boxplus y(x) = 2 \boxtimes b(x) \tag{2.16}$$

Actually, the algebraic relations (2.10), (2.11), (2.14), (2.15), and (2.16) are all obtained by using only the basic tools of probability and the definitions of addition and scalar multiplication in $\mathcal{P}_{\mathbb{F}_3}$*. We did not make use of the algebraic structure defined on* $\mathcal{P}_{\mathbb{F}_3}$ *to derive*

these relations. Further algebraic relations between the conditional pmfs defined in (2.9) can be obtained by using (2.10), (2.11), (2.14), (2.15), and (2.16) and exploiting the algebraic structure of $\mathcal{P}_{\mathbb{F}_3}$*. Some of these relations are given below.*

$$\begin{array}{lll} o(x) = -2 \boxtimes p(x) & y(x) = -2 \boxtimes g(x) & r(x) = -2 \boxtimes b(x) \\ p(x) = -\frac{1}{2} \boxtimes o(x) & g(x) = -\frac{1}{2} \boxtimes y(x) & b(x) = -\frac{1}{2} \boxtimes r(x) \end{array} \tag{2.17}$$

Example 2.2 *Since it is proven that* $\mathcal{P}_{\mathbb{F}_q}$ *is a linear vector space we can talk about linear mappings (transformations) from* $\mathcal{P}_{\mathbb{F}_q}$ *to other linear vector spaces. In this example we are going to provide a familiar example for such a mapping.*

The log-likelihood ratio (LLR), which is defined for binary valued pmfs as

$$\Lambda\{p(x)\} \triangleq \log \frac{p(0)}{p(1)}, \tag{2.18}$$

is a frequently employed tool in detection theory and channel decoding. For any $\alpha, \beta \in \mathbb{R}$ *and* $p(x), r(x) \in \mathcal{P}_{\mathbb{F}_2}$*,*

$$\begin{aligned} \Lambda\left\{\alpha \boxtimes p(x) \boxplus \beta \boxtimes r(x)\right\} &= \log \frac{C_{\mathbb{F}_2}\left\{(p(x))^{\alpha}(r(x))^{\beta}\right\}\Big|_{x=0}}{C_{\mathbb{F}_2}\left\{(p(x))^{\alpha}(r(x))^{\beta}\right\}\Big|_{x=1}} \\ &= \log \frac{(p(0))^{\alpha}(r(0)^{\beta})}{(p(1))^{\alpha}(r(1))^{\beta}} \\ &= \alpha\Lambda\{p(x)\} + \beta\Lambda\{r(x)\}. \end{aligned}$$

Hence, the LLR is a linear mapping from $\mathcal{P}_{\mathbb{F}_2}$ *to* $\mathbb{R}$*.*

2.5 The Geometric Structure over $\mathcal{P}_{\mathbb{F}_q}$

The geometric structure over a vector space is defined by means of an inner product. We are going to define an inner product on $\mathcal{P}_{\mathbb{F}_q}$ by first mapping the vectors of $\mathcal{P}_{\mathbb{F}_q}$ to $\mathbb{R}^q$ and then borrowing the usual inner product (dot product) on $\mathbb{R}^q$. Such a mapping should posses the properties stated in the following lemma.

Lemma 2.2 *Let* $\mathcal{M}\{.\}$ *be a mapping from* $\mathcal{P}_{\mathbb{F}_q}$ *to* $\mathbb{R}^q$ *and a function* $\sigma(.,.) : \mathcal{P}_{\mathbb{F}_q} \times \mathcal{P}_{\mathbb{F}_q} \to \mathbb{R}$ *be defined as*

$$\sigma(p(x), r(x)) \triangleq < \mathcal{M}\{p(x)\}, \mathcal{M}\{r(x)\} >_{\mathbb{R}^q}, \tag{2.19}$$

where $< ., . >_{\mathbb{R}^q}$ *denotes the usual inner product on* $\mathbb{R}^q$*.* $\sigma(p(x), r(x))$ *is an inner product on* $\mathcal{P}_{\mathbb{F}_q}$ *if* $\mathcal{M}\{.\}$ *is linear and injective (one-to-one).*

The proof of this lemma is given in Appendix A.1.1.

We propose the following mapping from $\mathcal{P}_{\mathbb{F}_q}$ to $\mathbb{R}^q$ and show later that it is linear and injective

$$\mathcal{L}\{p(x)\} \triangleq \sum_{i\in\mathbb{F}_q}\left(\log p(i) - \frac{1}{q}\sum_{j\in\mathbb{F}_q}\log p(j)\right)\mathbf{e}_i, \tag{2.20}$$

where $\mathbf{e}_i$ is the i^{th} canonical basis vector of $\mathbb{R}^q$[2]. The proposal for $\mathcal{L}\{.\}$ is inspired by the meaning of angle between two pmfs. The details of arriving at the definition of $\mathcal{L}\{.\}$ is given in Appendix A.1.2.

Lemma 2.3 *The mapping $\mathcal{L}\{.\} : \mathcal{P}_{\mathbb{F}_q} \to \mathbb{R}^q$ as defined in (2.20) is linear and injective.*

The proof is given Appendix A.1.3.

It is a common practice to map pmfs to log-probability vectors in the turbo decoding and sum-product algorithm literature. The main difference between those mappings and the mapping $\mathcal{L}\{.\}$ that we propose is the normalization $(-\frac{1}{q}\sum_{j\in\mathbb{F}_q}\log p(j)\,)$ in the definition of $\mathcal{L}\{.\}$. This normalization is necessary to make the operator $\mathcal{L}\{.\}$ linear and consequently allows us to borrow the inner product on $\mathbb{R}^q$. In other words, it is this normalization which allows us to construct a geometric structure on $\mathcal{P}_{\mathbb{F}_q}$. We believe that omitting this normalization in the literature hindered discovering the geometric relations between pmfs.

Obviously, the mapping $\mathcal{L}\{.\}$ is not the only mapping which satisfies the conditions imposed by Lemma 2.2. However, $\mathcal{L}\{.\}$ exhibits a symmetric form. This symmetry leads us to a useful geometric structure on $\mathcal{P}_{\mathbb{F}_q}$.

Theorem 2.4 *The function $< .,. >: \mathcal{P}_{\mathbb{F}_q} \times \mathcal{P}_{\mathbb{F}_q} \to \mathbb{R}$ defined for any $p(x), r(x) \in \mathcal{P}_{\mathbb{F}_q}$ as*

$$< p(x), r(x) > \stackrel{\triangle}{=} < \mathcal{L}\{p(x)\}, \mathcal{L}\{r(x)\} >_{\mathbb{R}^q}, \tag{2.21}$$

where $\mathcal{L}\{.\}$ is defined in (2.20), is an inner product on $\mathcal{P}_{\mathbb{F}_q}$.

The proof directly follows from Lemma 2.2 and Lemma 2.3.

[2] The canonical basis vectors of $\mathbb{R}^q$ are usually enumerated with integers from 1 up to q. In this thesis we enumerate the canonical basis vectors of $\mathbb{R}^q$ with the elements of $\mathbb{F}_q$. Since there are q canonical basis vectors of $\mathbb{R}^q$ and q elements in $\mathbb{F}_q$ there is not any problem in this enumeration.

The definition of the inner product on $\mathcal{P}_{\mathbb{F}_q}$ can be simplified as follows.

$$\begin{aligned} < p(x), r(x) > &= < \mathcal{L}\{p(x)\}, \mathcal{L}\{r(x)\} >_{\mathbb{R}^q} \\ &= \sum_{i\in\mathbb{F}_q}\left(\log p(i) - \frac{1}{q}\sum_{j\in\mathbb{F}_q}\log p(j)\right)\left(\log r(i) - \frac{1}{q}\sum_{j\in\mathbb{F}_q}\log r(j)\right) \quad (2.22) \\ &= \sum_{i\in\mathbb{F}_q}\log p(i)\log r(i) - \frac{1}{q}\left(\sum_{i\in\mathbb{F}_q}\log p(i)\right)\left(\sum_{i\in\mathbb{F}_q}\log r(i)\right) \quad (2.23) \end{aligned}$$

The equation above resembles the covariance of two random variables. Indeed, it is possible to express the definition of inner product in the form of a covariance of two real-valued random variables, which is shown in Appendix A.1.4.

The vector space $\mathcal{P}_{\mathbb{F}_q}$ evolves into an inner product space by the definition of the inner product in (2.21). Although we haven't shown what $\dim \mathcal{P}_{\mathbb{F}_q}$ is yet, we can conclude that $\mathcal{P}_{\mathbb{F}_q}$ is finite dimensional since there exist an injective mapping from $\mathcal{P}_{\mathbb{F}_q}$ to $\mathbb{R}^q$ [3]. It is well known from functional analysis theory that any finite dimensional inner product space is complete. Therefore, $\mathcal{P}_{\mathbb{F}_q}$ is a *Hilbert space*.

2.5.1 The norm, distance, and angle on $\mathcal{P}_{\mathbb{F}_q}$

The inner product on $\mathcal{P}_{\mathbb{F}_q}$ induces the following norm on $\mathcal{P}_{\mathbb{F}_q}$

$$\begin{aligned} \|p(x)\| &\triangleq \sqrt{< p(x), p(x) >} \quad (2.24) \\ &= \sqrt{\sum_{i\in\mathbb{F}_q}(\log p(i))^2 - \frac{1}{q}\left(\sum_{i\in\mathbb{F}_q}\log p(i)\right)^2}. \quad (2.25) \end{aligned}$$

A distance function between two pmfs can be obtained by combining this norm with the definition of subtraction in $\mathcal{P}_{\mathbb{F}_q}$ as in

$$\begin{aligned} D(p(x), r(x)) &\triangleq \|p(x) \boxminus r(x)\| \quad (2.26) \\ &= \sqrt{\sum_{i\in\mathbb{F}_q}\left(\log\frac{p(i)}{r(i)}\right)^2 - \frac{1}{q}\left(\sum_{i\in\mathbb{F}_q}\log\frac{p(i)}{r(i)}\right)^2}. \quad (2.27) \end{aligned}$$

Since $\|.\|$ is a proper norm, this distance is a metric distance. In other words, it is nonnegative, symmetric, and it satisfies the triangle equality.

[3] We are going to show that $\dim \mathcal{P}_{\mathbb{F}_q} = q - 1$ in Theorem 2.7

Similar to any Hilbert space, the angle between any two pmfs $p(x), r(x)$ in $\mathcal{P}_{\mathbb{F}_q}$ is given by

$$\angle(p(x), r(x)) \triangleq \arccos \frac{< p(x), r(x) >}{\|p(x)\| \, \|r(x)\|}. \tag{2.28}$$

2.5.2 The pseudo inverse of $\mathcal{L}\{.\}$

Lemma 2.5 *For any $p(x)$ in $\mathcal{P}_{\mathbb{F}_q}$*

$$\mathcal{L}\{p(x)\} \perp \mathbf{1}, \tag{2.29}$$

where $\mathbf{1}$ denotes the all one vector in $\mathbb{R}^q$.

The proof is given Appendix A.1.5.

Since $\mathcal{L}\{p(x)\}$ is always orthogonal to $\mathbf{1}$ it is not a surjection (onto). Consequently, it is not a bijection (injection and surjection). A mapping which is not a bijection does not have an inverse. Nonetheless, a pseudo inverse for $\mathcal{L}\{.\}$ exists which satisfies

$$\mathcal{L}^+\{\mathcal{L}\{p(x)\}\}(x) = p(x),$$

where $\mathcal{L}^+\{.\}(x)$ denotes the pseudo inverse of $\mathcal{L}\{.\}$.

$\mathcal{L}^+\{.\}(x)$ is a mapping from $\mathbb{R}^q$ to $\mathcal{P}_{\mathbb{F}_q}$. We propose the following definition for $\mathcal{L}^+\{.\}(x)$

$$\mathcal{L}^+\{\mathbf{p}\}(x) \triangleq C_{\mathbb{F}_q}\left\{\exp\left(-\frac{1}{2}\|\mathbf{p} - \mathbf{s}(x)\|^2\right)\right\}, \tag{2.30}$$

where $\mathbf{p}$ is any vector in $\mathbb{R}^q$ and $\mathbf{s}(x)$ is the vector-valued function from $\mathbb{F}_q$ to $\mathbb{R}^q$ given by

$$\mathbf{s}(x) \triangleq \mathbf{e}_x - \frac{1}{q}\mathbf{1}. \tag{2.31}$$

The definition of $\mathcal{L}^+\{.\}(x)$ can be interpreted as in

$$\mathcal{L}^+\{\mathbf{p}\}(x) = \Pr\{X = x | \mathbf{s}(X) + \mathbf{N} = \mathbf{p}\}, \tag{2.32}$$

where $\mathbf{N}$ is random vector whose components are all independent, real, zero-mean Gaussian random variables with unit variance. Furthermore, notice that the function $\mathbf{s}(x)$ maps the elements of $\mathbb{F}_q$ to $\mathbb{R}^q$ as in the simplex modulation.

Lemma 2.6 *$\mathcal{L}^+\{.\}(x) : \mathbb{R}^q \to \mathcal{P}_{\mathbb{F}_q}$ defined in (2.30) satisfies*

$$\mathcal{L}^+\{\mathcal{L}\{p(x)\}\}(x) = p(x) \tag{2.33}$$

for all $p(x)$ in $\mathcal{P}_{\mathbb{F}_q}$. Moreover,

$$\mathcal{L}\{\mathcal{L}^{+}\{\mathbf{p}\}(x)\} = \mathbf{p} \tag{2.34}$$

if $\mathbf{p} \perp \mathbf{1}$.

The proof is given in Appendix A.1.6.

Theorem 2.7 *$\mathcal{P}_{\mathbb{F}_q}$ is a $q-1$ dimensional Hilbert space, i.e..*

$$\dim \mathcal{P}_{\mathbb{F}_q} = q - 1 \tag{2.35}$$

Proof. Due to the rank-nullity theorem in linear algebra

$$\dim \mathcal{P}_{\mathbb{F}_q} = \dim \operatorname{im}\{\mathcal{L}\} + \dim \ker\{\mathcal{L}\},$$

where $\operatorname{im}\{\mathcal{L}\}$ and $\ker\{\mathcal{L}\}$ denote the image and kernel (null space) of $\mathcal{L}\{.\}$ respectively. Since $\mathcal{L}\{.\}$ is shown to be an injection in Lemma 2.3, $\ker\{\mathcal{L}\}$ only contains $\mathbf{0}$. It can be deduced from Lemma 2.5 that the image (range space) of $\mathcal{L}\{.\}$ is a subset of $\mathbf{1}^{\perp}$, where $\mathbf{1}^{\perp}$ is the subspace of $\mathbf{R}^q$ given by

$$\mathbf{1}^{\perp} \triangleq \{\mathbf{p} \in \mathbb{R}^q :< \mathbf{p}, \mathbf{1} >_{\mathbb{R}^q} = 0\} \tag{2.36}$$

The second part of Lemma 2.6 improves this result as it clearly shows that the image of $\mathcal{L}\{.\}$ is exactly equal to $\mathbf{1}^{\perp}$. Therefore,

$$\begin{aligned} \dim \mathcal{P}_{\mathbb{F}_q} &= \dim \mathbf{1}^{\perp} + \dim\{\mathbf{0}\} \\ &= q - 1, \end{aligned} \tag{2.37}$$

which completes the proof. ■

2.5.3 A set of orthonormal basis pmfs for $\mathcal{P}_{\mathbb{F}_q}$

A set of $q-1$ linearly independent vectors are necessary to form a basis for $\mathcal{P}_{\mathbb{F}_q}$. An orthonormal basis for $\mathcal{P}_{\mathbb{F}_q}$ can be obtained by finding a set of orthonormal vectors in $\mathbf{1}^{\perp}$ and then by mapping these vectors to $\mathcal{P}_{\mathbb{F}_q}$ via $\mathcal{L}^{+}\{.\}(x)$. Let $q-1$ vectors in $\mathbb{R}^q$ be defined as

$$\begin{array}{lcccccc} \mathbf{s}_1 & \triangleq [& \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 & \dots & 0 &] \\ \mathbf{s}_2 & \triangleq [& \frac{1}{\sqrt{6}} & \frac{1}{\sqrt{6}} & -\frac{2}{\sqrt{6}} & \dots & 0 &] \\ \vdots & & \vdots & \vdots & \vdots & \ddots & \vdots & \\ \mathbf{s}_{q-1} & \triangleq [& \frac{1}{\sqrt{q(q-1)}} & \frac{1}{\sqrt{q(q-1)}} & \frac{1}{\sqrt{q(q-1)}} & \dots & -\frac{q-1}{\sqrt{q(q-1)}} &] \end{array}. \tag{2.38}$$

Clearly, all of these vectors are all in $\mathbf{1}^{\perp}$ and they are all mutually orthonormal. $q-1$ pmfs in $\mathcal{P}_{\mathbb{F}_q}$ can be obtained by mapping these vectors to $\mathcal{P}_{\mathbb{F}_q}$ via $\mathcal{L}^{+}\{.\}(x)$ as follows.

$$s_i(x) \triangleq \mathcal{L}^{+}\{\mathbf{s}_i\}(x) \quad \text{for } i = 1, 2, \ldots, q-1. \tag{2.39}$$

Due to the definition of the inner product and the second part of Lemma 2.6,

$$\begin{aligned} < s_i(x), s_j(x) > &= < \mathcal{L}\{s_i(x)\}, \mathcal{L}\left\{s_j(x)\right\} >_{\mathbb{R}^q} \\ &= < \mathbf{s}_i, \mathbf{s}_j >_{\mathbb{R}^q} \\ &= \begin{cases} 1 & \text{, for } i = j \\ 0 & \text{, for } i \neq j \end{cases}. \end{aligned} \tag{2.40}$$

Therefore, $\{s_1(x), s_2(x), \ldots, s_{q-1}(x)\}$ is an orthonormal basis for $\mathcal{P}_{\mathbb{F}_q}$.

Example 2.3 *In Example 2.1 basic algebraic relations between six pmfs, which are in $\mathcal{P}_{\mathbb{F}_3}$, is investigated. An orthonormal basis for $\mathcal{P}_{\mathbb{F}_3}$ is composed of two pmfs. $s_1(x)$ and $s_2(x)$ given below forms such a basis for $\mathcal{P}_{\mathbb{F}_3}$.*

$$\begin{aligned} s_1(x) &= \mathcal{L}^{+}\left\{\begin{bmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 \end{bmatrix}^T\right\}(x) \\ &\simeq \begin{cases} 0.57598, & x = 0 \\ 0.14002, & x = 1 \\ 0.28400, & x = 2 \end{cases} \end{aligned} \tag{2.41}$$

$$\begin{aligned} s_2(x) &= \mathcal{L}^{+}\left\{\begin{bmatrix} \frac{1}{\sqrt{6}} & \frac{1}{\sqrt{6}} & -\frac{2}{\sqrt{6}} \end{bmatrix}^T\right\}(x) \\ &\simeq \begin{cases} 0.43595, & x = 0 \\ 0.43595, & x = 1 \\ 0.12810, & x = 2 \end{cases} \end{aligned} \tag{2.42}$$

The coordinates of a pmf in $\mathcal{P}_{\mathbb{F}_3}$, with respect to (w.r.t.) the basis $\{s_1(x), s_2(x)\}$ is simply the inner product of the pmf with $s_1(x)$ and $s_2(x)$. For instance, $r(x)$ mentioned in Example 2.1 can be expressed as

$$\begin{aligned} r(x) &= < r(x), s_1(x) > \boxtimes s_1(x) \boxplus < r(x), s_2(x) > \boxtimes s_2(x) \\ &\simeq -1.5537 \boxtimes s_1(x) \boxplus -0.89701 \boxtimes s_2(x). \end{aligned} \tag{2.43}$$

The coordinates of all the pmfs mentioned in Example 2.1 are given in the table below and depicted in Figure 2.2.

Table 2.2: Coordinates of the pmfs mentioned in Examples 2.1 and 2.3

	$r(x)$	$y(x)$	$o(x)$	$b(x)$	$g(x)$	$p(x)$
$s_1(x)$	-1.55367	1.55367	0	0.77684	-0.77684	0
$s_2(x)$	-0.89701	-0.89701	1.79403	0.44851	0.44851	-0.89701

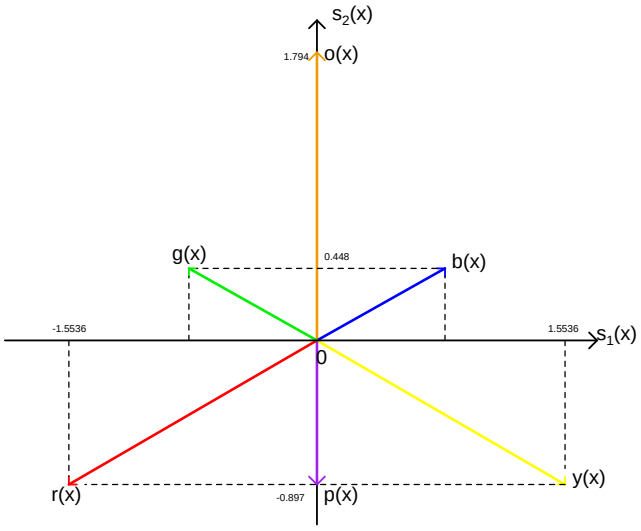

Figure 2.2: Plot of the pmfs mentioned in Examples 2.1 and 2.3

2.6 Relation to the Hilbert space of random variables

The Hilbert space of probability mass functions of finite field-valued random variables might be confused with the Hilbert space of random variables with a finite second order moment which is already well known [31]. However, these two Hilbert spaces are quite different from each other. First of all, the vectors of the former Hilbert space are pmfs of the random variables whereas the vectors of the latter Hilbert space are the random variables themselves. Second, the former Hilbert space is related to the finite-field valued random variables whereas the latter is related to the *complex-valued* random variables. Finally, the former is meaningful in the Bayesian detection sense whereas the latter is not.

Although this thesis is about the Hilbert space of the pmfs of finite field-valued random variables, it is adequate to summarize the Hilbert space of random variables. The set of *complex-valued* random variables forms vector space with the usual random variable addition and scaling over $\mathbb{C}$. This vector space can be endowed with the following inner product which is

nothing but the autocorrelation between two random variables.

$$< X, Y >= \mathbf{E}\left[XY^*\right], \tag{2.44}$$

where X and Y are two *complex-valued* random variables and $\mathbf{E}[.]$ denotes the expectation. The set of complex-valued random variables with finite second order moment is complete w.r.t. the norm induced by the inner product above. Therefore, this set forms a Hilbert space over $\mathbb{C}$ with the usual random variable addition, scaling, and the inner product given in (2.44). Many important algorithms, such as the Wiener filter, relies upon the orthogonality in this Hilbert space.

Notice that the Hilbert space structure over random variables is constructed over complex-valued random variables. Although it is also possible to construct a similar *vector space* over the set of finite field-valued random variables, the vector space of finite field-valued random variables does not have an inner product. In other words, the set of $\mathbb{F}_q$-valued random variables forms a vector space with the usual random variable addition and scaling over $\mathbb{F}_q$. On the contrary to complex-valued random variable case, the expected value is not a well defined concept for finite field-valued random variables. Consequently, autocorrelation between two $\mathbb{F}_q$-valued random variables is not well defined either. Therefore, we cannot construct a Hilbert space structure over the set of $\mathbb{F}_q$-valued random variables as we could for the complex valued random variables. If we had a Hilbert space structure over the set of $\mathbb{F}_q$-valued random variables then we would have decoding algorithms for linear channel codes with polynomial complexity.

2.6.1 Comparison between the convergence of random variables and pmfs

Another possible confusion might arise between the convergence of finite field-valued random variables and the convergence of pmfs of finite field-valued random variables. As explained above expectation is not well defined for finite field-valued random variables. Therefore, convergence in the mean square sense is not well defined for finite field-valued random variables either. On the other hand, convergence modes such as convergence almost everywhere and convergence in probability can still be well defined. However, due to the topological nature of the finite fields these two convergence modes are essentially equivalent. Convergence of a sequence of finite-field-valued random variables in probability is formally defined below.

Definition 2 Convergence of a sequence of finite-field-valued random variables in probability: *A sequence of $\mathbb{F}_q$-valued random variables, $\{X_n\}_{n=1}^{\infty}$, converges in probability to an $\mathbb{F}_q$-valued random variable X if and only if for each $\epsilon > 0$ there exist an integer N such that*

$$n > N \implies \Pr\{X_n = X\} > 1 - \epsilon \tag{2.45}$$

and this convergence is denoted by

$$\lim_{n\to\infty} \Pr\{X_n = X\} = 1. \tag{2.46}$$

Convergence of $\mathbb{F}_q$-valued random variables in probability, might be confused with the convergence of pmfs in $\mathcal{P}_{\mathbb{F}_q}$. The following example aims to clarify the distinction between these two convergences.

Example 2.4 *Let the event space of an experiment Ω be $[0, 1] \subset \mathbb{R}$ and each outcome of the experiment is equally likely, i.e.*

$$\Pr\{\omega \leq c\} = c, \tag{2.47}$$

where ω denotes the outcome of the experiment. A sequence of $\mathbb{F}_2$-valued random variables, $\{X_n\}_{n=1}^{\infty}$, are assigned to this experiment as follows.

$$X_n(\omega) \triangleq \begin{cases} 0, & \omega \in [0, 1 - 2^{-n}] \\ 1, & \omega \in (1 - 2^{-n}, 1] \end{cases} \tag{2.48}$$

Clearly, the sequence $\{X_n\}_{n=1}^{\infty}$ converges in probability to a random variable X which is defined as

$$X \triangleq \begin{cases} 0, & \omega \in [0, 1] \\ 1, & \omega \in \emptyset \end{cases} \tag{2.49}$$

In other words,

$$\lim_{n\to\infty} \Pr\{X_n = X\} = 1. \tag{2.50}$$

Let a sequence $\{p_n(x)\}_{n=1}^{\infty}$ of pmfs in $\mathcal{P}_{\mathbb{F}_2}$ be defined as

$$\begin{aligned} p_n(x) &\triangleq \Pr\{X_n = x\} \\ &= \begin{cases} 1 - 2^{-n}, & x = 0 \\ 2^{-n}, & x = 1 \end{cases} \end{aligned} \tag{2.51}$$

and $p(x)$ denote $\Pr\{X = x\}$. Due to the basic axioms of probability

$$p(x) = \begin{cases} 1 & , x = 0 \\ 0 & , x = 1 \end{cases} . \tag{2.52}$$

It might appear at a first glance that the sequence $\{p_n(x)\}_{n=1}^{\infty}$ converges to $p(x)$. However, this would contradict with the completeness of $\mathcal{P}_{\mathbb{F}_2}$ since $p(x) \notin \mathcal{P}_{\mathbb{F}_2}$. The truth is $\{p_n(x)\}_{n=1}^{\infty}$ is not a Cauchy sequence in $\mathcal{P}_{\mathbb{F}_2}$. This fact can be shown as follows. For any $m > n > 0$

$$\begin{aligned} D(p_m(x), p_n(x)) &= \sqrt{\sum_{i\in\mathbb{F}_2}\left(\log\frac{p_m(i)}{p_n(i)}\right)^2 - \frac{1}{q}\left(\sum_{i\in\mathbb{F}_2}\log\frac{p_m(i)}{p_n(i)}\right)^2} \\ &= \frac{1}{\sqrt{2}}\left(\log\frac{p_m(0)}{p_n(0)} + \log\frac{p_n(1)}{p_m(1)}\right). \end{aligned}$$

Since $p_m(0) > p_n(0)$

$$\begin{aligned} D(p_m(x), p_n(x)) &> \frac{1}{\sqrt{2}}\left(\log\frac{p_n(1)}{p_m(1)}\right) \\ &= \frac{\log 2}{\sqrt{2}}(m-n). \end{aligned} \tag{2.53}$$

Therefore, $\{p_n(x)\}_{n=1}^{\infty}$ is not a Cauchy sequence and the limit $\lim_{n\to\infty} p_n(x)$ does not exist. This example demonstrates that convergence of a sequence of random variables in probability does not imply the convergence of their pmfs.

2.7 The Hilbert space of multivariate pmfs

The construction of the Hilbert space on $\mathcal{P}_{\mathbb{F}_q}$ can be applied to the set of multivariate (joint) pmfs as well. Basically, we should replace the indeterminate variable x in the Hilbert space of pmfs with a vector $\mathbf{x}$ while constructing the Hilbert space structure on multivariate pmfs.

Let $\mathbf{X} = [X_1, X_2, \ldots, X_N]$ be a random vector where X_i is a $\mathbb{F}_q$-valued random variable. Furthermore, let $\mathcal{P}_{\mathbb{F}_q^N}$ denote the set of all strictly positive pmfs that $\mathbf{X}$ might posses, i.e.

$$\mathcal{P}_{\mathbb{F}_q^N} \triangleq \left\{ p(\mathbf{x}) : \mathbb{F}_q^N \to (0,1) \subset \mathbb{R}, \sum_{\mathbf{x}\in\mathbb{F}_q^N} p(\mathbf{x}) = 1 \right\}. \tag{2.54}$$

The addition and scalar multiplication on $\mathcal{P}_{\mathbb{F}_q^N}$ can be defined for any $p_1(\mathbf{x}), p_2(\mathbf{x}), p(\mathbf{x}) \in \mathcal{P}_{\mathbb{F}_q}$ and $\alpha \in \mathbb{R}$ as

$$p_1(\mathbf{x}) \boxplus p_2(\mathbf{x}) \triangleq C_{\mathbb{F}_q^N}\{p_1(\mathbf{x})p_2(\mathbf{x})\} \tag{2.55}$$

$$\alpha \boxtimes p(\mathbf{x}) \triangleq C_{\mathbb{F}_q^N}\{(p(\mathbf{x}))^{\alpha}\} \tag{2.56}$$

The normalization operator in the multivariate case, which is denoted by $C_{\mathbb{F}_q^N}\{.\}$ above, maps any strictly positive function of $\mathbb{F}_q^N$, $\alpha(\mathbf{x})$, to a pmf in $\mathcal{P}_{\mathbb{F}_q^N}$ as follows.

$$C_{\mathbb{F}_q^N}\{\alpha(\mathbf{x})\} \triangleq \frac{\alpha(\mathbf{x})}{\sum_{\mathbf{i}\in\mathbb{F}_q^N}\alpha(\mathbf{i})} \tag{2.57}$$

Similar to the univariate case, $\mathcal{P}_{\mathbb{F}_q^N}$ together with the $\boxplus$ and $\boxtimes$ operations forms a vector space over $\mathbb{R}$.

The analogue of the mapping $\mathcal{L}\{.\}$ in the multivariate case is denoted by $\mathcal{L}_N\{.\}$ and maps the pmfs in $\mathcal{P}_{\mathbb{F}_q^N}$ to $\mathbb{R}^{(q^N)}$. Before giving the definition of $\mathcal{L}_N\{.\}$ we need to establish a one-to-one matching between the *vectors* in $\mathbb{F}_q^N$ and the *canonical basis vectors* of $\mathbb{R}^{(q^N)}$. We can do this matching since $\mathbb{F}_q^N$ contains q^N vectors which is equal to the dimension of $\mathbb{R}^{(q^N)}$. Since the mapping $\mathcal{L}_N\{.\}$ is going to be employed in borrowing the inner product in $\mathbb{R}^{(q^N)}$ the order of matching is not important.

Using this matching $\mathcal{L}_N\{.\}$ is defined as

$$\mathcal{L}_N\{p(\mathbf{x})\} : \mathcal{P}_{\mathbb{F}_q^N} \to \mathbb{R}^{(q^N)} \triangleq \sum_{\mathbf{i}\in\mathbb{F}_q^N}\left(\log p(\mathbf{i}) - \frac{1}{q^N}\sum_{\mathbf{j}\in\mathbb{F}_q^N}\log p(\mathbf{j})\right)\mathbf{e_i}, \tag{2.58}$$

where $\mathbf{e_i}$ denotes the canonical basis vector of $\mathbb{R}^{(q^N)}$ matched to $\mathbf{i}\in\mathbb{F}_q^N$. $\mathcal{L}_N\{.\}$ is a linear and injective mapping as $\mathcal{L}\{.\}$. Then the inner product of any two $p(\mathbf{x}), r(\mathbf{x}) \in \mathcal{P}_{\mathbb{F}_q^N}$ becomes

$$< p(\mathbf{x}), r(\mathbf{x}) > \triangleq < \mathcal{L}_N\{p(\mathbf{x})\}, \mathcal{L}_N\{r(\mathbf{x})\} >_{\mathbb{R}^{(q^N)}} \tag{2.59}$$

$$= \sum_{\mathbf{i}\in\mathbb{F}_q^N}\log p(\mathbf{i})\log r(\mathbf{i}) - \frac{1}{q^N}\left(\sum_{\mathbf{i}\in\mathbb{F}_q^N}\log p(\mathbf{i})\right)\left(\sum_{\mathbf{i}\in\mathbb{F}_q^N}\log r(\mathbf{i})\right). \tag{2.60}$$

The definition of inner product makes $\mathcal{P}_{\mathbb{F}_q^N}$ an inner product space. Since $\mathcal{P}_{\mathbb{F}_q^N}$ is definitely finite dimensional it is also a Hilbert space.

The pseudo inverse of $\mathcal{L}_N\{.\}$ is

$$\mathcal{L}_N^+\{\mathbf{p}\}(\mathbf{x}) : \mathbb{R}^{(q^N)} \to \mathcal{P}_{\mathbb{F}_q^N} \triangleq C_{\mathbb{F}_q^N}\left\{\exp\left(-\frac{1}{2}\|\mathbf{p} - \mathbf{s}_N(\mathbf{x})\|^2\right)\right\}, \tag{2.61}$$

where $\mathbf{s}_N(\mathbf{x})$ is

$$\mathbf{s}_N(\mathbf{x}) \triangleq \mathbf{e_x} - \frac{1}{q^N}\mathbf{1}. \tag{2.62}$$

The vector $\mathbf{1}$ above denotes the all one vector in $\mathbb{R}^{(q^N)}$. Similar to the univariate case it can be shown that $\mathcal{L}_N^+\{.\}(\mathbf{x})$ satisfies

$$\mathcal{L}_N^+\{\mathcal{L}_N\{p(\mathbf{x})\}\}(\mathbf{x}) = p(\mathbf{x}) \qquad \forall p(\mathbf{x}) \in \mathcal{P}_{\mathbb{F}_q} \tag{2.63}$$

$$\mathcal{L}_N\left\{\mathcal{L}_N^+\{\mathbf{p}\}(\mathbf{x})\right\} = \mathbf{p} \qquad \forall \mathbf{p} \in \mathbf{1}^\perp \subset \mathbb{R}^{(q^N)}. \tag{2.64}$$

Consequently,

$$\mathrm{im}\,\{\mathcal{L}_N\} = \mathbf{1}^{\perp} \subset \mathbb{R}^{(q^N)} \tag{2.65}$$

Theorem 2.8 *$\mathcal{P}_{\mathbb{F}_q^N}$ is a $q^N - 1$ dimensional Hilbert space, i.e.*

$$\dim \mathcal{P}_{\mathbb{F}_q^N} = q^N - 1. \tag{2.66}$$

Proof. Due to the rank-nullity theorem in linear algebra

$$\begin{aligned} \dim \mathcal{P}_{\mathbb{F}_q^N} &= \dim \ker\{\mathcal{L}_N\} + \dim \mathrm{im}\,\{\mathcal{L}_N\} && (2.67)\\ &= q^N - 1. && (2.68) \end{aligned}$$

■

As a minor consequence of this theorem we can conclude that $\mathcal{P}_{\mathbb{F}_q^N}$ is isomorphic to $\mathcal{P}_{\mathbb{F}_{q^N}}$. This is a quite expected result since $\mathbb{F}_q^N$ is isomorphic to $\mathbb{F}_{q^N}$.

CHAPTER 3

THE CANONICAL FACTORIZATION OF MULTIVARIATE PROBABILITY MASS FUNCTIONS

3.1 Introduction

The factorization of a multivariate pmf is important in many aspects. For instance, the conditional dependence of the random variables distributed by a pmf can be determined by how the pmf factors. Existence of low complexity maximization and marginalization algorithms for a multivariate pmf, such as Viterbi and BCJR, also depends on the factorization of the pmf. A very special factorization of multivariate pmfs which we call as the canonical factorization is introduced in this chapter.

This chapter begins with representing the factorization of a pmf in $\mathcal{P}_{\mathbb{F}_q^N}$. Then we introduce the soft parity check constraints using which we decompose $\mathcal{P}_{\mathbb{F}_q^N}$ into orthogonal subspaces. Finally, we obtain the canonical factorization of pmfs as the projection of pmfs onto these subspaces.

3.2 Representing the factorization of pmfs

The Hilbert space $\mathcal{P}_{\mathbb{F}_q^N}$ provides a suitable environment for analyzing the factorization of multivariate pmfs. Suppose that a pmf in $\mathcal{P}_{\mathbb{F}_q^N}$ can be factored as

$$p(\mathbf{x}) = \prod_{i=1}^{K} \phi_i(\mathbf{x}). \tag{3.1}$$

Each $\phi_i(\mathbf{x})$ function appearing above may be called a factor function, a local function, a constraint, or an interaction. The factor functions are not necessarily pmfs but they can be as-

sumed to be positive. Hence, we can obtain a pmf in $\mathcal{P}_{\mathbb{F}_q}$ by scaling the factor functions as in

$$r_i(\mathbf{x}) = C_{\mathbb{F}_q^N}\{\phi_i(\mathbf{x})\} \tag{3.2}$$

$$= \frac{1}{\gamma_i}\phi_i(\mathbf{x}), \tag{3.3}$$

where $\gamma_i = \sum_{\mathbf{i}\in\mathbb{F}_q^N} \phi_i(\mathbf{x})$. After this normalization the factorization in (3.1) becomes

$$p(\mathbf{x}) = \prod_{i=1}^{K} \gamma_i r_i(\mathbf{x}) \tag{3.4}$$

$$= C_{\mathbb{F}_q^N}\left\{\prod_{i=1}^{K} r_i(\mathbf{x})\right\}, \tag{3.5}$$

which can be represented using the addition in $\mathcal{P}_{\mathbb{F}_q}$ as

$$p(\mathbf{x}) = \overset{\boxplus}{\sum_{i=1}^{K}} r_i(\mathbf{x}). \tag{3.6}$$

This representation suggests that a multivariate pmf in $\mathcal{P}_{\mathbb{F}_q^N}$ can be factored by expressing it as a linear combination of some basis vectors (pmfs) in $\mathcal{P}_{\mathbb{F}_q^N}$. If these basis pmfs are chosen to be orthogonal then we can employ the inner product on $\mathcal{P}_{\mathbb{F}_q^N}$ to determine the expansion coefficients. However, the basis pmfs should be selected in such a way that the resulting factorization becomes *useful*.

We know from the literature on the sum-product algorithm [1, 2, 6, 7] and Markov random fields [17, 18, 19, 20] that the factorization of $p(\mathbf{x})$ given in (3.1) is useful if the factor functions on the right hand side of (3.1) are *local*. A factor function of $p(\mathbf{x})$ is said to be local if it depends on some but not all of the components of the argument vector $\mathbf{x}$. Therefore, the basis functions mentioned in the paragraph above should also be selected to be as local as possible.

3.3 The multivariate pmfs that can be expressed as a function of a linear combination of their arguments

In this section we propose a special type of multivariate pmfs which will serve as basis vectors to obtain a factorization of pmfs in $\mathcal{P}_{\mathbb{F}_q}$. We show in the next chapter that the factorization obtained using these basis pmfs is quite useful. These basis pmfs are inspired by the parity check relations in $\mathbb{F}_q$. Suppose that the components of an $\mathbb{F}_q^N$-valued random vector $\mathbf{X} =$

$[X_1, X_2, \ldots, X_N]$ satisfy the following parity check relation

$$a_1X_1 + a_2X_2 + \ldots + a_NX_N = 0, \tag{3.7}$$

where a_i is a constant in $\mathbb{F}_q$. If all configurations satisfying this relation are assumed to be equiprobable then the joint pmf of $\mathbf{X}$, which is denoted by $p(\mathbf{x})$, is

$$p(\mathbf{x}) = \begin{cases} \frac{1}{q^{N-1}}, & \sum_{i=1}^{N} a_i x_i = 0 \\ 0, & \text{otherwise} \end{cases}. \tag{3.8}$$

This pmf can be expressed in a more compact form as

$$\begin{aligned} p(\mathbf{x}) &= \frac{1}{q^{N-1}}\delta(\mathbf{a}\mathbf{x}^T) && (3.9) \\ &= C_{\mathbb{F}_q^N}\left\{\delta(\mathbf{a}\mathbf{x}^T)\right\}, && (3.10) \end{aligned}$$

where $\mathbf{a}$ is $[a_1, a_2, \ldots, a_N]$ and $\delta(.)$ denotes the Kronecker delta.

The multivariate pmfs which can be expressed in the form as in (3.10) are called parity check or zero-sum constraints. A parity check constraint depends only on the variables which have nonzero coefficients associated with them. Hence, they posses local function properties as we desire from a basis pmf. Therefore, parity check constraints could be good candidates for being basis pmfs if they were elements of $\mathcal{P}_{\mathbb{F}_q^N}$. However, parity check constraints are not elements of $\mathcal{P}_{\mathbb{F}_q^N}$, since their value is zero for the configurations which do not satisfy the parity check relation.

We can obtained a "softened" version of the parity check constraints as follows. Suppose that the components of the random vector $\mathbf{X}$ satisfy the following relation instead of (3.7)

$$a_1X_1 + a_2X_2 + \ldots + a_NX_N = U, \tag{3.11}$$

where U is an $\mathbb{F}_q$-valued random variable distributed with an $r(u) \in \mathcal{P}_{\mathbb{F}_q}$. If all configurations resulting with the same value of U are assumed to be equiprobable then joint pmf of $\mathbf{X}$ in this case becomes

$$p(\mathbf{x}) = \begin{cases} \frac{1}{q^{N-1}}r(0), & \sum_{i=1}^{N} a_i x_i = 0 \\ \frac{1}{q^{N-1}}r(1), & \sum_{i=1}^{N} a_i x_i = 1 \\ \quad\vdots & \quad\vdots \\ \frac{1}{q^{N-1}}r(q-1), & \sum_{i=1}^{N} a_i x_i = q-1 \end{cases} \tag{3.12}$$

which can be expressed in a more compact form as

$$p(\mathbf{x}) = \frac{1}{q^{N-1}} r(\mathbf{a}\mathbf{x}^T) \tag{3.13}$$

$$= C_{\mathbb{F}_q^N}\left\{r(\mathbf{a}\mathbf{x}^T)\right\}. \tag{3.14}$$

Definition 3 *A multivariate pmf $p(\mathbf{x})$ in $\mathcal{P}_{\mathbb{F}_q^N}$ is called* a soft parity check (SPC) constraint *if there exist a $r(x) \in \mathcal{P}_{\mathbb{F}_q}$ and a vector $\mathbf{a} = [a_0, a_1, \ldots, a_{N-1}] \in \mathbb{F}_q^N$ such that*

$$p(\mathbf{x}) = C_{\mathbb{F}_q^N}\left\{r(\mathbf{a}\mathbf{x}^T)\right\}. \tag{3.15}$$

The vector $\mathbf{a}$ *is called the* parity check coefficient vector *of the SPC constraint $p(\mathbf{x})$.*

The difference between parity check and SPC constraint is the distribution of the weighted sum of the random variables X_0, X_1, …, X_{N-1}, which is denoted by U in (3.11). U is distributed with $\delta(u)$ in the parity check case whereas it is distributed with a $r(u)$ in $\mathcal{P}_{\mathbb{F}_q}$ in the SPC constraint case. The term "soft" arises from the fact that the weighted sum can take all values with some probability rather than guaranteed to be zero. Therefore, unlike parity check constraints SPC constraints are in $\mathcal{P}_{\mathbb{F}_q^N}$, since all configurations have nonzero probabilities.

Example 3.1 *Let two pmfs in $\mathcal{P}_{\mathbb{F}_3^2}$ are given with a slight abuse of notation as*

$$p_1(x_0, x_1) = \frac{1}{30}\begin{bmatrix} 3 & 6 & 1 \\ 6 & 1 & 3 \\ 1 & 3 & 6 \end{bmatrix}$$

$$p_2(x_0, x_1) = \frac{1}{157}\begin{bmatrix} 12 & 30 & 1 \\ 10 & 6 & 48 \\ 12 & 8 & 30 \end{bmatrix},$$

where $p_k(x_0 = i, x_1 = j)$ is given by the entry in the $(i+1)^{th}$ row and the $(j+1)^{th}$ column of the corresponding matrix.

Notice that $p_1(x_0, x_1)$ can be expressed as

$$p_1(x_0, x_1) = \frac{1}{3} r(x_0 + x_1)$$

$$= C_{\mathbb{F}_3^2}\{r(x_0 + x_1)\}$$

where $r(x) \in \mathcal{P}_{\mathbb{F}_3^N}$ *is*

$$r(x) = \begin{cases} 0.3, & x = 0 \\ 0.6, & x = 1 \\ 0.1 & x = 1 \end{cases}.$$

Therefore, $p_1(x_0, x_1)$ *is an SPC constraint with parity check coefficient vector* $[1, 1]$*. On the other hand, we cannot find a similar expression for* $p_2(x_0, x_1)$*. Hence,* $p_2(x_0, x_1)$ *is not an SPC constraint.*

Notice that we exploited the field structure of $\mathbb{F}_q$ in the discussion above. Parity check relations could also be described in finite rings but the number of configurations satisfying a parity check relation depends on the parity check coefficients in a finite ring. Therefore, the SPC constraints in a finite ring would not be in a nice form as above.

In the rest of this chapter we are going to show that SPC constraints form a complete set of orthogonal basis functions for $\mathcal{P}_{\mathbb{F}_q^N}$. The first step of this process is the following lemma which analyzes the inner product of two SPC constraints.

Lemma 3.1 Inner product of two SPC constraints: *Let* $p_1(\mathbf{x}), p_2(\mathbf{x}) \in \mathcal{P}_{\mathbb{F}_q^N}$ *are two SPC constraints such that*

$$\begin{aligned} p_1(\mathbf{x}) &= C_{\mathbb{F}_q^N}\left\{r_1(\mathbf{a}\mathbf{x}^T)\right\} \\ p_2(\mathbf{x}) &= C_{\mathbb{F}_q^N}\left\{r_2(\mathbf{b}\mathbf{x}^T)\right\}, \end{aligned} \tag{3.16}$$

where $r_1(x), r_2(x) \in \mathcal{P}_{\mathbb{F}_q}$*. If* $\mathbf{a}$ *and* $\mathbf{b}$ *are both nonzero vectors in* $\mathbb{F}_q^N$ *then*

$$< p_1(\mathbf{x}), p_2(\mathbf{x}) >= \begin{cases} q^{N-1} < r_1(x), r_2(\alpha x) >, & \exists \alpha \in \mathbb{F}_q : \mathbf{b} = \alpha \mathbf{a} \\ 0, & \textit{otherwise} \end{cases} \tag{3.17}$$

The proof of this lemma is given in Appendix A.2.1.

3.4 Orthogonal Subspace Decomposition of $\mathcal{P}_{\mathbb{F}_q^N}$

Generating an SPC constraint in $\mathcal{P}_{\mathbb{F}_q^N}$ based on a pmf in $\mathcal{P}_{\mathbb{F}_q}$ and a parity check coefficient vector $\mathbf{a}$ can be viewed as a mapping from $\mathcal{P}_{\mathbb{F}_q}$ to $\mathcal{P}_{\mathbb{F}_q^N}$ parameterized on $\mathbf{a}$ as given below.

$$\mathcal{S}_{\mathbf{a}}\left\{p(x)\right\} : \mathcal{P}_{\mathbb{F}_q} \rightarrow \mathcal{P}_{\mathbb{F}_q^N} \triangleq C_{\mathbb{F}_q^N}\left\{p(\mathbf{a}\mathbf{x}^T)\right\} \tag{3.18}$$

Any SPC constraint with parity check coefficient vector $\mathbf{a}$ is in $\mathrm{im}\{\mathcal{S}_{\mathbf{a}}\}$. The first of the following pair of lemmas states that $\mathrm{im}\{\mathcal{S}_{\mathbf{a}}\}$ is a subspace of $\mathcal{P}_{\mathbb{F}_q^N}$ and the second one investigates the relation between two such subspaces.

Lemma 3.2 *For any nonzero parity check coefficient vector* $\mathbf{a}$ *in* $\mathbb{F}_q^N$, $\mathrm{im}\{\mathcal{S}_{\mathbf{a}}\}$ *is a* $q-1$ *dimensional subspace of* $\mathcal{P}_{\mathbb{F}_q^N}$.

Lemma 3.3 *For any two nonzero parity check coefficient vectors* $\mathbf{a}, \mathbf{b} \in \mathbb{F}_q^N$

$$\exists \alpha \in \mathbb{F}_q : \mathbf{a} = \alpha \mathbf{b} \quad \Longrightarrow \quad \mathrm{im}\{\mathcal{S}_{\mathbf{a}}\} = \mathrm{im}\{\mathcal{S}_{\mathbf{b}}\} \tag{3.19}$$

$$\nexists \alpha \in \mathbb{F}_q : \mathbf{a} = \alpha \mathbf{b} \quad \Longrightarrow \quad \mathrm{im}\{\mathcal{S}_{\mathbf{a}}\} \perp \mathrm{im}\{\mathcal{S}_{\mathbf{b}}\} \tag{3.20}$$

The proofs of this lemmas are given in Appendix A.2.2 and Appendix A.2.3 respectively.

Lemma 3.3 suggests that $\mathcal{P}_{\mathbb{F}_q^N}$ can be decomposed into orthogonal subspaces by using a sufficient number of parity check coefficient vectors which are all pairwise linearly independent. Fortunately, we can borrow such a set of parity check coefficient vectors from coding theory as explained by the following theorem.

Theorem 3.4 *There exists a set* $\mathcal{H}$ *of pairwise linearly independent parity check vectors in* $\mathbb{F}_q$ *of length N such that*

$$\bigoplus_{\mathbf{a} \in \mathcal{H}} \mathrm{im}\{\mathcal{S}_{\mathbf{a}}\} = \mathcal{P}_{\mathbb{F}_q^N}. \tag{3.21}$$

where $\bigoplus$ *denotes orthogonal direct summation.*

Proof. For all nonzero $\mathbf{a}$, $\mathrm{im}\{\mathcal{S}_{\mathbf{a}}\}$ is a subspace of $\mathcal{P}_{\mathbb{F}_q^N}$. Orthogonal direct sum of subspaces is again a subspace of $\mathcal{P}_{\mathbb{F}_q^N}$. Therefore, we can complete the proof by finding an $\mathcal{H}$ which makes

$$\dim \bigoplus_{\mathbf{a} \in \mathcal{H}} \mathrm{im}\{\mathcal{S}_{\mathbf{a}}\} = \dim \mathcal{P}_{\mathbb{F}_q^N}. \tag{3.22}$$

Let the elements of $\mathcal{H}$ be selected by *transposing the columns* of the parity check matrix of the Hamming code in $\mathbb{F}_q$ with N rows. It is known from coding theory that the parity check matrix of such a Hamming code consists of $\frac{q^N-1}{q-1}$ columns all of which are pairwise linearly independent [11]. Therefore, $\mathcal{H}$ contains $\frac{q^N-1}{q-1}$ *pairwise linearly independent* vectors. Since these vectors are pairwise linearly independent, for any $\mathbf{a}, \mathbf{b} \in \mathcal{H}$

$$\mathrm{im}\{\mathcal{S}_{\mathbf{a}}\} \perp \mathrm{im}\{\mathcal{S}_{\mathbf{b}}\} \tag{3.23}$$

due to Lemma 3.3. Hence,

$$\dim \bigoplus_{\mathbf{a}\in\mathcal{H}} \operatorname{im}\{\mathcal{S}_{\mathbf{a}}\} = \sum_{\mathbf{a}\in\mathcal{H}} \dim \operatorname{im}\{\mathcal{S}_{\mathbf{a}}\}, \tag{3.24}$$

since these subspace are all orthogonal. $\operatorname{im}\{\mathcal{S}_{\mathbf{a}}\}$ is a $q-1$ dimensional subspace due to Lemma 3.2. Therefore,

$$\begin{aligned} \dim \bigoplus_{\mathbf{a}\in\mathcal{H}} \operatorname{im}\{\mathcal{S}_{\mathbf{a}}\} &= \sum_{\mathbf{a}\in\mathcal{H}} (q-1) \\ &= |\mathcal{H}|(q-1) \\ &= q^N - 1 \\ &= \dim \mathcal{P}_{\mathbb{F}_q^N}, \end{aligned} \tag{3.25}$$

which completes the proof. ■

3.5 The Canonical Factorization

Corollary 3.5 (The fundamental result of the thesis:) *Any multivariate pmf in $\mathcal{P}_{\mathbb{F}_q^N}$ can be expressed as a product of functions that depend on a linear combination of their arguments.*

Proof. Let $\mathcal{H}$ be set of parity check vectors satisfying (3.21), existence of which is guaranteed by Theorem 3.4. Let the vectors in $\mathcal{H}$ be enumerated as $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_{|\mathcal{H}|}$. Then any $p(\mathbf{x}) \in \mathcal{P}_{\mathbb{F}_q^N}$ can be expressed as

$$p(\mathbf{x}) = \boxplus \sum_{i=1}^{|\mathcal{H}|} p_i(\mathbf{x}) \tag{3.26}$$

where $p_i(\mathbf{x})$ is the projection of $p(\mathbf{x})$ onto $\operatorname{im}\{\mathcal{S}_{\mathbf{a}_i}\}$. Since $p_i(\mathbf{x})$ is in $\operatorname{im}\{\mathcal{S}_{\mathbf{a}_i}\}$, there exist an $r_i(x) \in \mathcal{P}_{\mathbb{F}_q}$ such that

$$p_i(\mathbf{x}) = C_{\mathbb{F}_q^N}\left\{r_i(\mathbf{a}_i\mathbf{x}^T)\right\}. \tag{3.27}$$

Then $p(\mathbf{x})$ can be expressed as

$$p(\mathbf{x}) = \boxplus \sum_{i=1}^{|\mathcal{H}|} C_{\mathbb{F}_q^N}\left\{r_i(\mathbf{a}_i\mathbf{x}^T)\right\}. \tag{3.28}$$

Employing the definition of addition in $\mathcal{P}_{\mathbb{F}_q^N}$ yields the desired factorization.

$$p(\mathbf{x}) = C_{\mathbb{F}_q^N}\left\{\prod_{i=1}^{|\mathcal{H}|} r_i(\mathbf{a}_i\mathbf{x}^T)\right\} \tag{3.29}$$

$$= \frac{1}{\gamma}\prod_{i=1}^{|\mathcal{H}|} r_i(\mathbf{a}_i\mathbf{x}^T), \tag{3.30}$$

where γ is equal to $\sum_{\forall \mathbf{i} \in \mathbb{F}_q^N} \prod_{i=1}^{|\mathcal{H}|} r_i(\mathbf{a}_i \mathbf{x}^T)$. ■

Definition 4 The canonical factorization: *A factorization of a multivariate pmf is called the canonical factorization of the pmf if all factor functions are SPC factors and parity check coefficient vectors of all SPC factors are pairwise linearly independent.*

The canonical factorization of a multivariate pmf in $\mathcal{P}_{\mathbb{F}_q^N}$ can be obtained by projecting the pmf onto the subspaces $\text{im}\{\mathcal{S}_{\mathbf{a}_i}\}$ for $\mathbf{a}_i \in \mathcal{H}$. In order to compute this projection a set of orthonormal basis pmfs for $\text{im}\{\mathcal{S}_{\mathbf{a}_i}\}$ is required. We can derive such a set of orthonormal basis pmfs from the orthonormal basis pmfs for $\mathcal{P}_{\mathbb{F}_q}$ given in Section 2.5.3 by using the first part of Lemma 3.1. The inner product of two SPC constraints $C_{\mathbb{F}_q^N}\left\{s_j(\mathbf{a}_i \mathbf{x}^T)\right\}$ and $C_{\mathbb{F}_q^N}\left\{s_k(\mathbf{a}_i \mathbf{x}^T)\right\}$ which are derived from $s_j(x)$ and $s_k(x)$ defined in (2.39) is

$$< C_{\mathbb{F}_q^N}\left\{s_j(\mathbf{a}_i \mathbf{x}^T)\right\}, C_{\mathbb{F}_q^N}\left\{s_k(\mathbf{a}_i \mathbf{x}^T)\right\} >= q^{N-1} < s_j(x), s_k(x) > \tag{3.31}$$

due to Lemma 3.1. Consequently,

$$< C_{\mathbb{F}_q^N}\left\{s_j(\mathbf{a}_i \mathbf{x}^T)\right\}, C_{\mathbb{F}_q^N}\left\{s_k(\mathbf{a}_i \mathbf{x}^T)\right\} >= \begin{cases} q^{N-1}, & k = j \\ 0 \end{cases} \tag{3.32}$$

Therefore, the set given below is a set of orthonormal basis pmfs for $\text{im}\{\mathcal{S}_{\mathbf{a}_i}\}$.

$$\left\{q^{-\frac{N-1}{2}} \boxtimes C_{\mathbb{F}_q^N}\left\{s_1(\mathbf{a}_i \mathbf{x}^T)\right\}, q^{-\frac{N-1}{2}} \boxtimes C_{\mathbb{F}_q^N}\left\{s_2(\mathbf{a}_i \mathbf{x}^T)\right\}, \ldots, q^{-\frac{N-1}{2}} \boxtimes C_{\mathbb{F}_q^N}\left\{s_{q-1}(\mathbf{a}_i \mathbf{x}^T)\right\}\right\} \tag{3.33}$$

Then the projection of $p(\mathbf{x})$ onto $\text{im}\{\mathcal{S}_{\mathbf{a}_i}\}$, which is denoted by $C_{\mathbb{F}_q^N}\left\{r_i(\mathbf{a}_i \mathbf{x}^T)\right\}$, can be obtained as

$$C_{\mathbb{F}_q^N}\left\{r_i(\mathbf{a}_i \mathbf{x}^T)\right\} = \overset{\boxplus}{\sum_{j=1}^{q-1}} q^{-(N-1)} \boxtimes < C_{\mathbb{F}_q^N}\left\{s_j(\mathbf{a}_i \mathbf{x}^T)\right\}, p(\mathbf{x}) > \boxtimes C_{\mathbb{F}_q^N}\left\{s_j(\mathbf{a}_i \mathbf{x}^T)\right\}. \tag{3.34}$$

Moreover, due to the linearity of the mapping $\mathcal{S}_{\mathbf{a}_i}\{.\}$

$$r_i(x) = \overset{\boxplus}{\sum_{j=1}^{q-1}} q^{-(N-1)} \boxtimes < C_{\mathbb{F}_q^N}\left\{s_j(\mathbf{a}_i \mathbf{x}^T)\right\}, p(\mathbf{x}) > \boxtimes s_j(x). \tag{3.35}$$

Example 3.2 *Suppose that we are required to find the canonical factorization of $p_2(x_0, x_1)$ given in Example 3.1. We can decompose $\mathcal{P}_{\mathbb{F}_3^2}$ into orthogonal subspaces with a set $\mathcal{H}$ containing $\frac{3^2-1}{3-1} = 4$ pairwise linearly independent parity check vectors of length two. Such an $\mathcal{H}$ can be selected as*

$$\mathcal{H} = \{[1, 0], [0, 1], [1, 1], [1, 2]\} \tag{3.36}$$

The subspaces of $\mathcal{P}_{\mathbb{F}_3^2}$ based on these parity check vectors are

$$\begin{aligned}
\operatorname{im}\{\mathcal{S}_{[1,0]}\} &= \left\{ p(x_0,x_1) = \frac{1}{3} r(x_0) = \frac{1}{3} \begin{bmatrix} r(0) & r(0) & r(0) \\ r(1) & r(1) & r(1) \\ r(2) & r(2) & r(2) \end{bmatrix} : r(x) \in \mathcal{P}_{\mathbb{F}_3} \right\} \\
\operatorname{im}\{\mathcal{S}_{[0,1]}\} &= \left\{ p(x_0,x_1) = \frac{1}{3} r(x_1) = \frac{1}{3} \begin{bmatrix} r(0) & r(1) & r(2) \\ r(0) & r(1) & r(2) \\ r(0) & r(1) & r(2) \end{bmatrix} : r(x) \in \mathcal{P}_{\mathbb{F}_3} \right\}
\end{aligned}$$

$$\begin{aligned}
\operatorname{im}\{\mathcal{S}_{[1,1]}\} &= \left\{ p(x_0,x_1) = \frac{1}{3} r(x_0+x_1) = \frac{1}{3} \begin{bmatrix} r(0) & r(1) & r(2) \\ r(1) & r(2) & r(0) \\ r(2) & r(1) & r(0) \end{bmatrix} : r(x) \in \mathcal{P}_{\mathbb{F}_3} \right\} \\
\operatorname{im}\{\mathcal{S}_{[1,2]}\} &= \left\{ p(x_0,x_1) = \frac{1}{3} r(x_0+2x_1) = \frac{1}{3} \begin{bmatrix} r(0) & r(1) & r(2) \\ r(2) & r(0) & r(1) \\ r(1) & r(2) & r(0) \end{bmatrix} : r(x) \in \mathcal{P}_{\mathbb{F}_3} \right\}
\end{aligned}$$

Let the projections of $p_2(x_0, x_1)$ onto these subspaces be denoted with $\frac{1}{3} r_1(x_0)$, $\frac{1}{3} r_2(x_1)$, $\frac{1}{3} r_3(x_0 + x_1)$, and $\frac{1}{3} r_4(x_0 + 2x_1)$ respectively. These pmfs can be computed using (3.35) as

$$r_1(x) = \begin{cases} 0.2, & x = 0 \\ 0.4, & x = 1 \\ 0.4, & x = 2 \end{cases}, \quad r_2(x) = \begin{cases} \frac{1}{3}, & x = 0 \\ \frac{1}{3}, & x = 1 \\ \frac{1}{3}, & x = 2 \end{cases}$$

$$r_3(x) = \begin{cases} 0.4, & x = 0 \\ 0.5, & x = 1 \\ 0.1, & x = 2 \end{cases}, \quad r_4(x) = \begin{cases} 0.3, & x = 0 \\ 0.6, & x = 1 \\ 0.1, & x = 2 \end{cases}.$$

Finally, it can be verified that

$$p_2(x_0, x_1) = \frac{1500}{157} r_1(x_0) r_2(x_1) r_3(x_0 + x_1) r_4(x_0 + 2x_1).$$

CHAPTER 4

PROPERTIES AND SPECIAL CASES OF THE CANONICAL FACTORIZATION

4.1 Introduction

The canonical factorization deserves its name by possessing some important properties. This chapter explains these properties first and then some special cases of the canonical factorization is derived. These special cases will be important while applying the canonical factorization to communication theory problems in Chapter 6. This chapter begins with introducing a matrix notation to represent local functions in Section 4.2. Then it is shown in Section 4.3 that the canonical factorization is the ultimate factorization possible. The uniqueness of the canonical factorization is explained Section 4.4. The canonical factorization of pmfs with known alternative factorizations is derived in Section 4.5. This chapter ends with deriving the canonical factorization of the joint pmf a random vector obtained by linear transformation of another random vector.

4.2 Representation of local functions

In the rest of the thesis we deal frequently with local functions. We adopt a matrix notation to indicate the variables that a factor function depends. We use $\mathbb{F}_q$-valued diagonal matrices such that some of their entries on the main diagonal are 1 and the rest are all 0. For instance,

$$p(\mathbf{x}) = p(\mathbf{x}\mathbf{D}) \tag{4.1}$$

indicates that the pmf $p(\mathbf{x})$ depends on only to the components of $\mathbf{x}$ associated with a 1 on the diagonal of the matrix $\mathbf{D}$. We call such matrices dependency matrices. Some special

dependency matrices we use in the thesis are $\mathbf{E}_i$, $\mathbf{I}$, and $\mathbf{O}$. $\mathbf{E}_i$ denotes the dependency matrix with a 1 only on the i^{th} entry of its diagonal. The other two matrices are the identity matrix and the all-zeros matrix respectively.

A local pmf is orthogonal to some SPC constraints as shown by the following lemma. This lemma is quite useful not only in this chapter but also in Chapter 7.

Lemma 4.1 *For any $p(\mathbf{x}) \in \mathcal{P}_{\mathbb{F}_q^N}$, any nonzero $\mathbf{a} \in \mathbb{F}_q^N$, and any dependency matrix $\mathbf{D}$*

$$p(\mathbf{x}) = p(\mathbf{x}\mathbf{D}) \wedge \mathbf{a}\mathbf{D} \neq \mathbf{a} \implies p(\mathbf{x}) \perp \operatorname{im}\{\mathcal{S}_{\mathbf{a}}\}. \tag{4.2}$$

The proof is given in Appendix A.3.1.

4.3 Ultimateness of the canonical factorization

The ultimate goal of any mathematical factorization operation is to factor the mathematical object to its most basic building blocks. For instance, the goal of integer factorization is to express a natural number as a product of prime numbers. Similarly, the ultimate goal of polynomial factorization is to express a polynomial as a product of irreducible polynomials.

In the case of factoring a strictly positive multivariate pmf into strictly positive factor functions, it is difficult to set an ultimate goal or to describe the most basic building blocks of multivariate pmfs. Since, any factor function in any factorization can still be expressed as a product of other positive factor functions, a multivariate pmf can be factored arbitrarily in many different ways and the factorization operation can continue indefinitely. In this aspect, factoring a strictly positive pmf is similar to trying to factor a real number.

However, not every factorization is useful in practice. A factorization of a multivariate pmf is useful if it expresses the pmf as a product of *local* functions. Therefore, it is reasonable to continue to factor a multivariate pmf if any factor function can still be expressed as a product of *more local* factor functions. For instance, let a factor function $\phi(\mathbf{x}\mathbf{D})$ of $p(\mathbf{x})$ be expressed as

$$\phi(\mathbf{x}\mathbf{D}) = \phi_1(\mathbf{x}\mathbf{D}_1)\phi_2(\mathbf{x}\mathbf{D}_2)$$

where $\mathbf{D}_i \neq \mathbf{D}$ but $\mathbf{D}\mathbf{D}_i = \mathbf{D}_i$ for $i = 1, 2$. Since $\phi_1(.)$ and $\phi_2(.)$ have less number of arguments than $\phi(.)$ has, the ultimate factorization of $p(\mathbf{x})$ should contain the product $\phi_1(\mathbf{x}\mathbf{D}_1)\phi_2(\mathbf{x}\mathbf{D}_2)$

rather than $\phi(\mathbf{xD})$. In this point of view, the canonical factorization is the ultimate factorization that one can achieve as stated by the following theorem.

Theorem 4.2 *An SPC factor function with a nonzero norm cannot be factored further to functions having less number of arguments.*

Proof. Assume that an SPC constraint, $C_{\mathbb{F}_q^N}\left\{r(\mathbf{a}\mathbf{x}^T)\right\}$, with a nonzero norm can be factored to functions having less number of arguments. In other words, assume that $C_{\mathbb{F}_q^N}\left\{r(\mathbf{a}\mathbf{x}^T)\right\}$ can be expressed as

$$\begin{aligned} C_{\mathbb{F}_q^N}\left\{r(\mathbf{a}\mathbf{x}^T)\right\} &= \phi_1(\mathbf{xD}_1)\phi_2(\mathbf{xD}_2) \\ &= C_{\mathbb{F}_q^N}\left\{\phi_1(\mathbf{xD}_1)\right\} \boxplus C_{\mathbb{F}_q^N}\left\{\phi_2(\mathbf{xD}_2)\right\}, \end{aligned} \tag{4.3}$$

where $\mathbf{D}_1$ and $\mathbf{D}_2$ are such dependency matrices that $\mathbf{aD}_1 \neq \mathbf{a}$ and $\mathbf{aD}_2 \neq \mathbf{a}$. $C_{\mathbb{F}_q^N}\left\{\phi_1(\mathbf{xD}_1)\right\}$ and $C_{\mathbb{F}_q^N}\left\{\phi_2(\mathbf{xD}_2)\right\}$ are orthogonal to $C_{\mathbb{F}_q^N}\left\{r(\mathbf{a}\mathbf{x}^T)\right\}$ due to Lemma 4.1. Then (4.3) is only possible if

$$C_{\mathbb{F}_q^N}\left\{r(\mathbf{a}\mathbf{x}^T)\right\} = C_{\mathbb{F}_q^N}\left\{\phi_1(\mathbf{xD}_1)\right\} = C_{\mathbb{F}_q^N}\left\{\phi_2(\mathbf{xD}_2)\right\} = \theta(\mathbf{x}),$$

which is a contradiction completing the proof. ■

4.4 Uniqueness of the canonical factorization

Recall that we need a set $\mathcal{H}$ composed of $\frac{q^N-1}{q-1}$ pairwise linearly independent vectors in $\mathbb{F}_q^N$ to derive the canonical factorization of a pmf in $\mathcal{P}_{\mathbb{F}_q^N}$. There are $2^N - 1$ nonzero vectors in $\mathbb{F}_2^N$ all of which are pairwise linearly independent. Hence, the set $\mathcal{H}$ should contain all the nonzero vectors in $\mathbb{F}_2^N$. Consequently, the set $\mathcal{H}$ required in the derivation of the canonical factorization of a pmf in $\mathcal{P}_{\mathbb{F}_2^N}$ is unique. Moreover, the canonical factorization obtained from such a set $\mathcal{H}$ is also unique.

If the $\mathbb{F}_q$ is not the binary field then there are $q^N - 1$ nonzero vectors in $\mathbb{F}_q^N$. Hence, we can have more than one distinct sets which contain $\frac{q^N-1}{q-1}$ pairwise linearly independent vectors in $\mathbb{F}_q^N$ if q is not equal to two. Let $\mathcal{H}_1 = \{\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_M\}$ and $\mathcal{H}_2 = \{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_M\}$ be two distinct sets containing $M = \frac{q^N-1}{q-1}$ pairwise linearly independent vectors in $\mathbb{F}_q^N$. Using these two sets we can obtain two different canonical factorizations of a multivariate pmf $p(\mathbf{x}) \in \mathcal{P}_{\mathbb{F}_q^N}$

as in

$$p(\mathbf{x}) = C_{\mathbb{F}_q^N}\left\{\prod_{i=1}^{M} r_i(\mathbf{a}_i\mathbf{x}^T)\right\}, \tag{4.4}$$

$$p(\mathbf{x}) = C_{\mathbb{F}_q^N}\left\{\prod_{i=1}^{M} t_i(\mathbf{b}_i\mathbf{x}^T)\right\}, \tag{4.5}$$

where $r_i(\mathbf{a}_i\mathbf{x}^T)$ and $t_i(\mathbf{b}_i\mathbf{x}^T)$ denote the projections of $p(\mathbf{x})$ onto $\mathrm{im}\{\mathcal{S}_{\mathbf{a}_i}\}$ and $\mathrm{im}\{\mathcal{S}_{\mathbf{b}_i}\}$ respectively. Notice that any $\mathbf{a}_i$ in $\mathcal{H}_1$ is definitely linearly dependent with one of the $\mathbf{b}_i$ vectors in $\mathcal{H}_2$. In other words for any $\mathbf{a}_i \in \mathcal{H}_1$ there exist a $\mathbf{b}_j \in \mathcal{H}_2$ such that

$$\mathbf{b}_j = \alpha\mathbf{a}_i. \tag{4.6}$$

Consequently, $\mathrm{im}\{\mathcal{S}_{\mathbf{a}_i}\}$ is equal to $\mathrm{im}\left\{\mathcal{S}_{\mathbf{b}_j}\right\}$ due to Lemma 3.3. Therefore, the projection of $p(\mathbf{x})$ onto these same subspaces should also be equal, i.e.,

$$r_i(\mathbf{a}_i\mathbf{x}^T) = t_j(\mathbf{b}_j\mathbf{x}^T), \tag{4.7}$$

which means that the factorizations in (4.4) and (4.5) are essentially the same factorization although they appear different. Since different sets of parity check coefficient vectors leads to the same canonical factorization, we can conclude that the canonical factorization of a given pmf is unique. Since the selection of the vectors in $\mathcal{H}$ does not affect the resulting canonical factorization, in the rest of the thesis we use $\mathcal{H}$ to denote any set containing $\frac{q^N-1}{q-1}$ pairwise linearly independent vectors in $\mathbb{F}_q^N$.

4.5 The canonical factorization of pmfs with alternative factorizations

In the most general case, the canonical factorization of a multivariate pmf in $\mathcal{P}_{\mathbb{F}_q^N}$ is composed of $|\mathcal{H}|$ SPC factors. However, for some special pmfs some of these $|\mathcal{H}|$ SPC factors are essentially constants. For these pmfs less than $|\mathcal{H}|$ SPC factors may suffice to express the canonical factorization.

The first group of these special types of pmfs consists of pmfs which depend on only a subset of their arguments. The canonical factorization of these types of pmfs is investigated in the following lemma.

Lemma 4.3 *Let $\mathcal{D}$ be a subset of $\mathcal{H}$ defined for a dependency matrix* $\mathbf{D}$ *as*

$$\mathcal{D} \triangleq \{\mathbf{a}_i \in \mathcal{H} : \mathbf{a}_i\mathbf{D} = \mathbf{a}_i\}. \tag{4.8}$$

The canonical factorization of a multivariate pmf $p(\mathbf{x}) \in \mathcal{P}_{\mathbb{F}_q}$ is in the form of

$$p(\mathbf{x}) = C_{\mathbb{F}_q^N}\left\{\prod_{\mathbf{a}_i \in \mathcal{D}} r_i(\mathbf{a}_i \mathbf{x}^T)\right\} \tag{4.9}$$

if and only if

$$p(\mathbf{x}) = p(\mathbf{x}\mathbf{D}). \tag{4.10}$$

Proof. Due to Theorem 3.4 any pmf in $\mathcal{P}_{\mathbb{F}_q^N}$ can be expressed as

$$\begin{aligned} p(\mathbf{x}) &= \overset{\boxplus}{\sum_{\mathbf{a}_i \in \mathcal{H}}} C_{\mathbb{F}_q^N}\left\{r_i(\mathbf{a}_i \mathbf{x}^T)\right\} && (4.11)\\ &= \overset{\boxplus}{\sum_{\mathbf{a}_i \in \mathcal{D}}} C_{\mathbb{F}_q^N}\left\{r_i(\mathbf{a}_i \mathbf{x}^T)\right\} \boxplus \overset{\boxplus}{\sum_{\mathbf{a}_i \in \mathcal{H}\setminus\mathcal{D}}} C_{\mathbb{F}_q^N}\left\{r_i(\mathbf{a}_i \mathbf{x}^T)\right\}, && (4.12)\end{aligned}$$

where $C_{\mathbb{F}_q^N}\left\{r_i(\mathbf{a}_i \mathbf{x}^T)\right\}$ is the projection of $p(\mathbf{x})$ onto $\operatorname{im}\{\mathcal{S}_{\mathbf{a}}\}$. But $p(\mathbf{x})$ is orthogonal to $\operatorname{im}\{\mathcal{S}_{\mathbf{a}_i}\}$ for $\mathbf{a}_i \in \mathcal{H} \setminus \mathcal{D}$ due to Lemma 4.1. Hence,

$$C_{\mathbb{F}_q^N}\left\{r_i(\mathbf{a}_i \mathbf{x}^T)\right\} = \theta(\mathbf{x}), \tag{4.13}$$

for $\mathbf{a}_i \in \mathcal{H} \setminus \mathcal{D}$. Consequently,

$$\begin{aligned} p(\mathbf{x}) &= \overset{\boxplus}{\sum_{\mathbf{a}_i \in \mathcal{D}}} C_{\mathbb{F}_q^N}\left\{r_i(\mathbf{a}_i \mathbf{x}^T)\right\} && (4.14)\\ &= C_{\mathbb{F}_q^N}\left\{\prod_{\mathbf{a}_i \in \mathcal{D}} r_i(\mathbf{a}_i \mathbf{x}^T)\right\}, && (4.15)\end{aligned}$$

which is the desired factorization to prove the theorem in the forward direction.

The proof in the backward direction is straight forward. If $p(\mathbf{x})$ can be factored as in (4.9) then

$$\begin{aligned} p(\mathbf{x}\mathbf{D}) &= C_{\mathbb{F}_q^N}\left\{\prod_{\mathbf{a}_i \in \mathcal{D}} r_i(\mathbf{a}_i \mathbf{x}^T)\right\}\Bigg|_{\mathbf{x}=\mathbf{x}\mathbf{D}} && (4.16)\\ &= C_{\mathbb{F}_q^N}\left\{\prod_{\mathbf{a}_i \in \mathcal{D}} r_i(\mathbf{a}_i \mathbf{D}^T \mathbf{x}^T)\right\}. && (4.17)\end{aligned}$$

Since $\mathbf{D}$ is symmetric and $\mathbf{a}_i\mathbf{D} = \mathbf{a}_i$ for $\mathbf{a}_i \in \mathcal{D}$,

$$\begin{aligned} p(\mathbf{x}\mathbf{D}) &= C_{\mathbb{F}_q^N}\left\{\prod_{\mathbf{a}_i \in \mathcal{D}} r_i(\mathbf{a}_i \mathbf{x}^T)\right\} && (4.18)\\ &= p(\mathbf{x}), && (4.19)\end{aligned}$$

which completes the proof. ■

This lemma tells in practice that any pmf satisfying the relation $p(\mathbf{x}) = p(\mathbf{xD})$ can be expressed as a product of $|\mathcal{D}|$ SPC factors rather than $|\mathcal{H}|$ SPC factors. Moreover, parity check coefficient vectors of these SPC factors satisfy the relation $\mathbf{a} = \mathbf{aD}$. We do not need to compute the projection of $p(\mathbf{x})$ onto $\text{im}\{\mathcal{S}_{\mathbf{a}}\}$ if $\mathbf{a}$ is not in $\mathcal{D}$, since the result of that projection would be $\theta(\mathbf{x})$ definitely.

The next theorem investigates the canonical factorization of pmfs with known alternative factorizations.

Theorem 4.4 *If a multivariate pmf $p(\mathbf{x}) \in \mathcal{P}_{\mathbb{F}_q^N}$ can be factored as*

$$p(\mathbf{x}) = C_{\mathbb{F}_q^N}\left\{\prod_{j=1}^{K} \phi_j(\mathbf{xD}_j)\right\}, \tag{4.20}$$

where $\mathbf{D}_1$, $\mathbf{D}_2$, ..., $\mathbf{D}_K$ are dependency matrices then the canonical factorization of $p(\mathbf{x})$ is in the form of

$$p(\mathbf{x}) = C_{\mathbb{F}_q^N}\left\{\prod_{j=1}^{K} \prod_{\mathbf{a}_i \in \mathcal{D}_j} r_i(\mathbf{a}_i\mathbf{x}^T)\right\}, \tag{4.21}$$

where $\mathcal{D}_j$ is the subset of $\mathcal{H}$ given by

$$\mathcal{D}_j \triangleq \{\mathbf{a}_i \in \mathcal{H} : \mathbf{a}_i\mathbf{D}_j = \mathbf{a}_i\}. \tag{4.22}$$

Proof. This theorem is actually a direct consequence of Lemma 4.3. Let $t_j(\mathbf{x}) \in \mathcal{P}_{\mathbb{F}_q^N}$ be

$$t_j(\mathbf{x}) \triangleq C_{\mathbb{F}_q^N}\left\{\phi_j(\mathbf{xD}_j)\right\}. \tag{4.23}$$

Since $t_j(\mathbf{x})$ is equal to $t_j(\mathbf{xD}_j)$,

$$t_j(\mathbf{x}) = \boxplus\sum_{\mathbf{a}_i \in \mathcal{D}_j} r_i(\mathbf{a}_i\mathbf{x}^T), \tag{4.24}$$

due to Lemma 4.3. Then $p(\mathbf{x})$ is

$$\begin{aligned} p(\mathbf{x}) &= \boxplus\sum_{j=1}^{K} t_j(\mathbf{x}) && (4.25)\\ &= \boxplus\sum_{j=1}^{K} \boxplus\sum_{\mathbf{a}_i \in \mathcal{D}_j} r_i(\mathbf{a}_i\mathbf{x}^T) && (4.26)\\ &= C_{\mathbb{F}_q^N}\left\{\prod_{j=1}^{K} \prod_{\mathbf{a}_i \in \mathcal{D}_j} r_i(\mathbf{a}_i\mathbf{x}^T)\right\}, && (4.27) \end{aligned}$$

which completes the proof. ■

The practical consequence of this theorem is that the canonical factorization of a pmf with an alternative factorization can be derived by obtaining the canonical factorization of the factor functions in the alternative factorization. This approach significantly simplifies the derivation of the canonical factorization for such pmfs and extensively used in Chapter 6.

4.6 The effect of reversible linear transformations on the canonical factorization

If two random vectors are related with a *reversible* linear transformation then the canonical factorization of the pmf of the one of random vectors can be derived from the canonical factorization of the other random vector's pmf. Let $\mathbf{X}$ be an $\mathbb{F}_q^N$-valued random vector distributed with $p(\mathbf{x}) \in \mathcal{P}_{\mathbb{F}_q^N}$. Moreover, let $\mathbf{Y}$ be another $\mathbb{F}_q^N$-valued random vector which is related to $\mathbf{X}$ as in

$$\mathbf{Y} = \mathbf{X}\mathbf{B} \tag{4.28}$$

where $\mathbf{B}$ is an *reversible* matrix in $\mathbb{F}_q^{N\times N}$. Since $\mathbf{B}$ is reversible, for each $\mathbf{y} \in \mathbb{F}_q^N$ there is one and only one $\mathbf{x} \in \mathbb{F}_q^N$ vector satisfying $\mathbf{y} = \mathbf{x}\mathbf{B}$, which is given by $\mathbf{x} = \mathbf{y}\mathbf{B}^{-1}$. Hence,

$$\begin{aligned} \Pr\{\mathbf{Y} = \mathbf{y}\} &= \Pr\{\mathbf{X} = \mathbf{y}\mathbf{B}^{-1}\} && (4.29)\\ &= p(\mathbf{y}\mathbf{B}^{-1}). && (4.30) \end{aligned}$$

If the canonical factorization of $p(\mathbf{x})$ is as given in

$$p(\mathbf{x}) = \prod_{\mathbf{a}_i \in \mathcal{H}} r_i(\mathbf{a}_i \mathbf{x}^T) \tag{4.31}$$

then the canonical factorization of $\Pr\{\mathbf{Y} = \mathbf{y}\}$ is simply

$$\begin{aligned} \Pr\{\mathbf{Y} = \mathbf{y}\} &= \prod_{\mathbf{a}_i \in \mathcal{H}} r_i(\mathbf{a}_i \mathbf{x}^T)\Big|_{\mathbf{x}=\mathbf{y}\mathbf{B}^{-1}} && (4.32)\\ &= \prod_{\mathbf{a}_i \in \mathcal{H}} r_i\left(\mathbf{a}_i (\mathbf{B}^{-1})^T \mathbf{y}^T\right) && (4.33) \end{aligned}$$

If $\mathbf{B}$ was not reversible then $\Pr\{\mathbf{Y} = \mathbf{y}\}$ would be zero for some $\mathbf{y}$ vectors in $\mathbb{F}_q^N$. Hence, $\Pr\{\mathbf{Y} = \mathbf{y}\}$ would not be a multivariate pmf in $\mathcal{P}_{\mathbb{F}_q^N}$ and consequently we could not talk about the canonical factorization of $\Pr\{\mathbf{Y} = \mathbf{y}\}$.

An interesting question about the linear transformations of $\mathbb{F}_q^N$-valued random vectors might be whether there exists a linear transformation $\mathbf{K}$ for a random vector $\mathbf{X}$ such that the components of the vector $\mathbf{Y} = \mathbf{XK}$ are statistically independent. If such a transformation exists it would prove useful in computing the marginal pmfs of the components of the random vector $\mathbf{X}$. The canonical factorization of $\Pr\{\mathbf{Y} = \mathbf{y}\}$ given in (4.33) provides a clue to this question.

Theorem 4.5 *There exist a matrix* $\mathbf{K}$ *in* $\mathbb{F}_q^{N\times N}$ *for an* $\mathbb{F}_q^N$*-valued random vector* $\mathbf{X}$ *such that the components of the random vector* $\mathbf{Y}$ *given by*

$$\mathbf{Y} = \mathbf{XK} \tag{4.34}$$

are ***statistically independent*** *if the canonical factorization of the pmf of* $\mathbf{X}$ *is composed of at most N SPC factors whose parity check coefficient vectors are all* ***linearly independent****.*

Proof. The proof is constructive. Let $p(\mathbf{x})$ be the pmf of $\mathbf{X}$ and the canonical factorization of $p(\mathbf{x})$ be denoted as

$$p(\mathbf{x}) = C_{\mathbb{F}_q^N}\left\{\prod_{\mathbf{a}_i\in\mathcal{K}} r_i(\mathbf{a}_i\mathbf{x}^T)\right\} \tag{4.35}$$

where $\mathcal{K}$ is a subset of $\mathcal{H}$ containing at most N linearly independent vectors. Let $\mathcal{K}_c$ be a subset of $\mathcal{H}$ such that it is a superset of $\mathcal{K}$ and it contains exactly N linearly independent vectors. Since $r_i(\mathbf{a}_i\mathbf{x}^T)$ is equal to $\theta(\mathbf{x})$ for $\mathbf{a}_i \in \mathcal{K}_c \setminus \mathcal{K}$, the canonical factorization of $p(\mathbf{x})$ can also be expressed as

$$p(\mathbf{x}) = C_{\mathbb{F}_q^N}\left\{\prod_{\mathbf{a}_i\in\mathcal{K}_c} r_i(\mathbf{a}_i\mathbf{x}^T)\right\} \tag{4.36}$$

We may define a matrix $\mathbf{K}_c$ whose rows are the elements of $\mathcal{K}_c$. Using this matrix $\mathbf{K}_c$ the canonical factorization of $p(\mathbf{x})$ becomes

$$p(\mathbf{x}) = C_{\mathbb{F}_q^N}\left\{\prod_{j=1}^{N} r_{i(j)}(\mathbf{f}_j\mathbf{K}_c\mathbf{x}^T)\right\}, \tag{4.37}$$

where $\mathbf{f}_j$ is the j^{th} canonical basis vector of $\mathbb{F}_q^N$ and $i(j)$ is the index of the vector $\mathbf{a}_i$ when $\mathbf{a}_i = \mathbf{f}_j\mathbf{K}_c$. Then we may define the matrix $\mathbf{K}$ as

$$\mathbf{K} \triangleq (\mathbf{K}_c^{-1})^T. \tag{4.38}$$

With this definition of $\mathbf{K}$, the canonical factorization of $\Pr\{\mathbf{Y} = \mathbf{y}\}$ becomes

$$\begin{aligned}
\Pr\{\mathbf{Y} = \mathbf{y}\} &= C_{\mathbb{F}_q^N}\left\{\prod_{j=1}^{N} r_{i(j)}(\mathbf{f}_j\mathbf{K}_c\mathbf{x}^T)\right\}\Bigg|_{\mathbf{x}=\mathbf{y}\mathbf{K}^{-1}} && (4.39)\\
&= C_{\mathbb{F}_q^N}\left\{\prod_{j=1}^{N} r_{i(j)}\left(\mathbf{f}_j\mathbf{K}_c(\mathbf{K}^{-1})^T\mathbf{y}^T\right)\right\} && (4.40)\\
&= C_{\mathbb{F}_q^N}\left\{\prod_{j=1}^{N} r_{i(j)}(\mathbf{f}_j\mathbf{y}^T)\right\} && (4.41)\\
&= C_{\mathbb{F}_q^N}\left\{\prod_{j=1}^{N} r_{i(j)}(y_j)\right\}, && (4.42)
\end{aligned}$$

where y_j is the j^{th} component of $\mathbf{y}$. Since $\Pr\{\mathbf{Y} = \mathbf{y}\}$ is separable, the components of $\mathbf{Y}$ are statistically independent. Moreover, the distribution of the j^{th} component of $\mathbf{Y}$ is simply

$$\Pr\{Y_j = y\} = r_{i(j)}(y). \tag{4.43}$$

■

In the general case, the marginal pmfs of the components of an $\mathbb{F}_q^N$-valued random vector $\mathbf{X}$ can be computed via the marginalization sum whose complexity is q^N. If the multivariate pmf of $\mathbf{X}$ obeys the condition imposed in Theorem 4.5 then $\mathbf{X}$ can be related to $\mathbf{Y}$, whose components are statistically independent, as

$$\mathbf{X} = \mathbf{Y}\mathbf{K}^{-1}. \tag{4.44}$$

This means that any component of $\mathbf{X}$ is equal to a linear combination of N statistically independent random variables. Hence, the marginal pmfs of the components of $\mathbf{X}$ can be computed via $N-1$ circular convolutions over $\mathbb{F}_q$ instead of the marginalization sum. Consequently, the complexity of computing a single marginal pmf is Nq^2 and the complexity of computing all marginal pmfs is N^2q^2 instead of q^N for such random vectors[1].

[1] These complexities can be reduced even more to $Nq\log_2 q$ and $N^2q\log_2 q$ by computing the convolutions via FFT if $\mathbb{F}_q$ is an extension field of the binary field [10].

CHAPTER 5

EMPLOYING CHANNEL DECODERS FOR INFERENCE TASKS BEYOND DECODING

5.1 Introduction

This chapter explains subjectively the most important consequence of the canonical factorization which allows the decoders of the linear error correction codes to be utilized in other inference tasks.

This chapter starts with an overview of channel decoders. Then how a maximum likelihood (ML) decoder can be used to maximize a multivariate pmf is explained. It is shown in Section 5.4 that symbolwise decoders can be employed to marginalize multivariate pmfs. Section 5.5 highlights that the decoders of the dual Hamming code can be used as universal inference machines. Special cases are analyzed in Section 5.6. The material presented in this chapter is summarized with graphical models in 5.7. This chapter ends with explaining the possible applications of employing channel decoders for inference tasks beyond decoding.

5.2 An overview of channel decoders

A channel decoder is specified by a code and a channel through which the coded symbols are transmitted. A code C over a finite field $\mathbb{F}_q$ of length L is defined as a subset of $\mathbb{F}_q^L$. The code is called a *linear code* if C is a subspace of $\mathbb{F}_q^L$. For linear codes there exists a matrix $\mathbf{H}$ which satisfies

$$\mathbf{H}\mathbf{x}^T = \mathbf{0} \quad \forall \mathbf{x} \in C. \tag{5.1}$$

The matrix $\mathbf{H}$ is called the parity check matrix of the code.

A channel is a system which maps a $\mathbb{F}_q$-valued symbol to an element of the output alphabet in a probabilistic manner [1]. We assume that the channel decoders used in the rest of this chapter are designed for a specific channel. This channel relates the inputs to the outputs via the following relation

$$\mathbf{Y}_i = \mathbf{s}(X_i) + \mathbf{Z}_i, \tag{5.2}$$

where $\mathbf{Z}_i$ is a noise vector consisting of independent, zero-mean, real Gaussian random variables with unit variance and $\mathbf{s}(.)$ denotes the simplex mapping as defined in (2.31). The likelihood function, which is a conditional probability density function of a continuous random vector, of this channel is

$$\begin{aligned} f_{\mathbf{Y}_i|X_i}\{\mathbf{Y}_i = \mathbf{y}_i | X_i = x_i\} &\propto \exp\left(-\frac{1}{2}\left\|\mathbf{y}_i - \mathbf{s}(x_i)\right\|^2\right) && (5.3)\\ &\propto \mathcal{L}^+\left\{\mathbf{y}_i\right\}(x_i). && (5.4) \end{aligned}$$

The reasoning behind the selection of this channel model is explained in Section 5.6.2.

Let $\mathbf{X} = [X_1, X_2, \ldots, X_L]$ denote a codeword belonging to the code C and $\mathbf{Y} = [\mathbf{Y}_1, \mathbf{Y}_2, \ldots, \mathbf{Y}_L]$ denote the output of the channel when $\mathbf{X}$ is transmitted through this channel. If all codewords are equally likely then the a posteriori probability (APP) of $\mathbf{X}$ is

$$\begin{aligned} \Pr\{\mathbf{X} = \mathbf{x} | \mathbf{Y} = \mathbf{y}\} &= C_{\mathbb{F}_q^L}\left\{\mathbb{1}_C(\mathbf{x}) \prod_{i=1}^{L} f_{\mathbf{Y}_i|X_i}\{\mathbf{Y}_i = \mathbf{y}_i | X_i = x_i\}\right\} && (5.5)\\ &= C_{\mathbb{F}_q^L}\left\{\mathbb{1}_C(\mathbf{x}) \prod_{i=1}^{L} \mathcal{L}^+\left\{\mathbf{y}_i\right\}(x_i)\right\}, && (5.6) \end{aligned}$$

where $\mathbf{x} = [x_1, x_2, \ldots, x_L]$, $\mathbf{y} = [\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_L]$, and $\mathbb{1}_C(.)$ denotes the indicator function i.e.,

$$\mathbb{1}_C(\mathbf{x}) \triangleq \begin{cases} 1, & \mathbf{x} \in C \\ 0, & \mathbf{x} \notin C \end{cases}. \tag{5.7}$$

If the code C is a linear code with the parity check matrix $\mathbf{H}$ consisting of M rows then

$$\mathbb{1}_C(\mathbf{x}) = \prod_{i=1}^{M} \delta(\mathbf{h}_i \mathbf{x}^T), \tag{5.8}$$

where $\mathbf{h}_i$ denotes the i^{th} row of $\mathbf{H}$. Consequently, the APP of $\mathbf{X}$ is

$$\Pr\{\mathbf{X} = \mathbf{x} | \mathbf{Y} = \mathbf{y}\} = C_{\mathbb{F}_q^L}\left\{\prod_{i=1}^{M} \delta(\mathbf{h}_i \mathbf{x}^T) \prod_{i=1}^{L} \mathcal{L}^+\left\{\mathbf{y}_i\right\}(x_i)\right\}. \tag{5.9}$$

[1] This definition of channel includes the modulator when necessary.

There are two decoding problems that can be associated with a code and the channel model defined above [33]. The first one of these decoding problems is the *codeword* decoding problem which is the task of inferring the transmitted codeword. This task is accomplished by finding the codeword which maximizes the APP $\Pr\{\mathbf{X} = \mathbf{x}|\mathbf{Y} = \mathbf{y}\}$. Hence, this decoding is called the maximum a posteriori (MAP) codeword decoding. The MAP codeword decoding can be formally defined as

$$\hat{\mathbf{x}}_{MAP} \triangleq \arg\max_{\mathbf{x}\in C} \Pr\{\mathbf{X} = \mathbf{x}|\mathbf{Y} = \mathbf{y}\}. \tag{5.10}$$

If all codewords are equally likely then the MAP codeword decoding problem is equal to the maximum likelihood (ML) codeword decoding problem which maximizes the likelihood function $f_{\mathbf{Y}|\mathbf{X}}\{\mathbf{Y} = \mathbf{y}|\mathbf{X} = \mathbf{x}\}$ instead of the APP, i.e.,

$$\begin{aligned} \hat{\mathbf{x}}_{ML} &\triangleq \arg\max_{\mathbf{x}\in C} f_{\mathbf{Y}|\mathbf{X}}\{\mathbf{Y} = \mathbf{y}|\mathbf{X} = \mathbf{x}\} && (5.11)\\ &= \arg\max_{\mathbf{x}\in C} \Pr\{\mathbf{X} = \mathbf{x}|\mathbf{Y} = \mathbf{y}\} && (5.12)\\ &= \hat{\mathbf{x}}_{MAP}. && (5.13) \end{aligned}$$

Both MAP and ML codeword decoding problems can be solved by the min-sum (max-product) algorithm, the most famous example of which is the Viterbi algorithm [2, 6, 7, 33].

The second decoding problem is the *symbolwise* decoding problem which aims to produce a soft prediction about the individual coded symbols. This task is accomplished by marginalizing the APP as in

$$\Pr\{X_i = x_i|\mathbf{Y} = \mathbf{y}\} = \sum_{\sim\{x_i\}} \Pr\{\mathbf{X} = \mathbf{x}|\mathbf{Y} = \mathbf{y}\} \tag{5.14}$$

where $\sim\{x_i\}$ is the summary notation introduced in [1] and indicates that the summation runs over the variables x_1, x_2, …, x_{i-1}, x_{i+1}, x_{i+2}, …, x_N. The symbolwise decoding problem is solved by the sum-product algorithm whose most famous example is the BCJR algorithm [24, 33].

5.3 Maximizing a multivariate pmf by using an ML codeword decoder

We begin this section with the following example. This example might be impractical but it is the simplest possible example to demonstrate the idea. We will generalize the idea after this example.

Example 5.1 *Suppose that we are required to implement a device which finds the configuration maximizing a pmf* $p(x_1, x_2) \in \mathcal{P}_{\mathbb{F}_2^2}$. *This device is supposed to return the pair* (x_1, x_2) *which maximizes* $p(x_1, x_2)$ *after receiving the values* $p(0,0)$, $p(0,1)$, $p(1,0)$, *and* $p(1,1)$ *as input. Assume that while implementing this device we can use a* handicapped *processor which can only add two numbers, negate a number, and compute the logarithm of a number but cannot compare two numbers. Further assume that to compensate the handicap of the processor we are given the ML codeword decoder hardware of the linear code with the parity check matrix*

$$\mathbf{H} = [\ 1 \quad 1 \quad 1\ ], \tag{5.15}$$

which is designed for the channel model described in Section 5.2.

If the processor at our hand was a regular processor which could compare two numbers then the solution of this problem would be obvious. Since this processor cannot compare two numbers, we need to figure out another solution by employing the ML codeword decoder. In this solution we should use the processor to compute the three input vectors[2] *to be applied to the decoder from inputs applied to the whole system.*

We sketch a solution as follows. Let the input vectors applied to the decoder be $\mathbf{y}_1$, $\mathbf{y}_2$, *and* $\mathbf{y}_3$. *By (5.9) this decoder will return the following* $\hat{\mathbf{x}}_{ML} = [\hat{x}_1, \hat{x}_2, \hat{x}_3]$ *vector*

$$\hat{\mathbf{x}}_{ML} = \arg \max_{[x_1,x_2,x_3]\in C} \delta(x_1 + x_2 + x_3) \prod_{i=1}^{3} \mathcal{L}^+ \{\mathbf{y}_i\}(x_i). \tag{5.16}$$

Since $x_3 = x_1 + x_2$ *for every codeword in* C,

$$\hat{\mathbf{x}}_{ML} = \arg \max_{[x_1,x_2,x_3]\in C} \mathcal{L}^+ \{\mathbf{y}_1\}(x_1)\mathcal{L}^+ \{\mathbf{y}_2\}(x_2)\mathcal{L}^+ \{\mathbf{y}_3\}(x_1 + x_2). \tag{5.17}$$

Due to Corollary 3.5 we know that any $p(x_1, x_2) \in \mathcal{P}_{\mathbb{F}_2^2}$ *can be expressed as*

$$p(x_1, x_2) = C_{\mathbb{F}_2^2} \{r_1(x_1)r_2(x_2)r_3(x_1 + x_2)\}. \tag{5.18}$$

Hence, if we apply $\mathbf{y}_i = \mathcal{L}\{r_i(x)\}$ *to the decoder then the decoder computes*

$$\begin{aligned} \hat{\mathbf{x}}_{ML} &= \arg \max_{[x_1,x_2,x_3]\in C} r_1(x_1)r_2(x_2)r_3(x_1 + x_2) && (5.19)\\ &= \arg \max_{[x_1,x_2,x_3]\in C} p(x_1, x_2). && (5.20) \end{aligned}$$

The first two components of the $\hat{\mathbf{x}}_{ML}$ *is the result we are looking for.*

[2] Recall that the channel model given in (5.2) maps each bit to a vector in $\mathbb{R}^2$

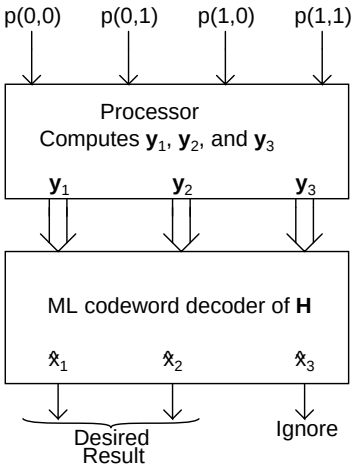

Figure 5.1: The block diagram of the solution to problem in Example 5.1.

The only missing component of the solution is computing $\mathbf{y}_i = \mathcal{L}\{r_i(x)\}$. *These vectors can be derived using the discussion in Chapter 3 as*

$$\begin{aligned}
\mathbf{y}_1 &= [\ \log p(0,0) + \log p(0,1) - \log p(1,0) - \log p(1,1)\ ][\ 1 \quad -1\ ] \\
\mathbf{y}_2 &= [\ \log p(0,0) - \log p(0,1) + \log p(1,0) - \log p(1,1)\ ][\ 1 \quad -1\ ] \\
\mathbf{y}_3 &= [\ \log p(0,0) - \log p(0,1) - \log p(1,0) + \log p(1,1)\ ][\ 1 \quad -1\ ].
\end{aligned} \tag{5.21}$$

Fortunately, our handicapped processor can be programmed to accomplish this subtask. The block diagram of the solution is depicted in Figure 5.1.

An ML codeword decoder can be utilized to maximize a multivariate pmf $t(\mathbf{x})$ if it can be expressed as a product of parity-check (zero-sum) constraints and degree one factors as in

$$t(\mathbf{x}) = \prod_{i=1}^{M} \delta(\mathbf{h}_i \mathbf{x}^T) \prod_{i=1}^{L} \phi_i(x_i). \tag{5.22}$$

The decoder which can maximize this pmf is the decoder of the linear code with the parity check matrix $\mathbf{H}$ given by

$$\mathbf{H} = \begin{bmatrix} \mathbf{h}_1 \\ \mathbf{h}_2 \\ \vdots \\ \mathbf{h}_M \end{bmatrix}. \tag{5.23}$$

If $\mathbf{y}_i = \mathcal{L}\left\{C_{\mathbb{F}_q^N}\{\phi_i(x)\}\right\}$ is applied as the i^{th} input to the decoder then the ML codeword decoder performs the following maximization

$$\begin{aligned}\hat{\mathbf{x}}_{ML} &= \arg\max_{\mathbf{x}\in C} \prod_{i=1}^{M} \delta(\mathbf{h}_i\mathbf{x}^T) \prod_{i=1}^{L} \mathcal{L}^+\{\mathbf{y}_i\}(x_i) && (5.24)\\ &= \arg\max_{\mathbf{x}} \prod_{i=1}^{M} \delta(\mathbf{h}_i\mathbf{x}^T) \prod_{i=1}^{L} \phi_i(x_i) && (5.25)\\ &= \arg\max_{\mathbf{x}} t(\mathbf{x}), && (5.26)\end{aligned}$$

which is the desired maximization.

Unfortunately, most of the pmfs cannot be factored as in (5.22). Therefore, it might seem that utilization of an ML codeword decoder for maximizing a pmf has limited applicability. However, for any pmf in $\mathcal{P}_{\mathbb{F}_q^N}$ we can find a substitute pmf which factors as in (5.22) and can be used to maximize the original pmf. Consequently, ML codeword decoders can be utilized in the maximization of a broad range of pmfs.

We derive such a substitute pmf based on the canonical factorization. Let $p(\mathbf{x}) \in \mathcal{P}_{\mathbb{F}_q^N}$ be the multivariate which we want to maximize by using an ML codeword decoder. Due to Corollary 3.5, $p(\mathbf{x})$ can be expressed as a product of SPC constraints as in

$$p(\mathbf{x}) = C_{\mathbb{F}_q^N}\left\{\prod_{i=1}^{|\mathcal{H}|} r_i(\mathbf{a}_i\mathbf{x}^T)\right\}, \tag{5.27}$$

where $\mathcal{H}$ is $\{\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_{|\mathcal{H}|}\}$ and $C_{\mathbb{F}_q^N}\left\{r_i(\mathbf{a}_i\mathbf{x}^T)\right\}$ is the projection of $p(\mathbf{x})$ onto $\text{im}\{\mathcal{S}_{\mathbf{a}_i}\}$. Recall that the set $\mathcal{H}$ consists of $\frac{q^N-1}{q-1}$ pairwise linearly independent parity check vectors. Since all of these parity check coefficient vectors are pairwise linearly independent, N of them have to be of weight one. Without loss of generality we may assume that these weight one vectors are the first N parity check vectors in $\mathcal{H}$, i.e. $\mathbf{a}_1$, $\mathbf{a}_2$, …, $\mathbf{a}_N$. Then we may define a matrix $\mathbf{A}$ by using the remaining parity check coefficient vectors in $\mathcal{H}$ as

$$\mathbf{A} \triangleq \begin{bmatrix} \mathbf{a}_{N+1} \\ \mathbf{a}_{N+2} \\ \vdots \\ \mathbf{a}_{|\mathcal{H}|} \end{bmatrix}. \tag{5.28}$$

We will use $\mathbf{A}$ while defining the substitute pmf for $p(\mathbf{x})$. This substitute pmf has an extended argument vector $\mathbf{x}_E$ consisting of L components where L is equal to $|\mathcal{H}|$. This extended argument vector is defined as

$$\mathbf{x}_E \triangleq [\mathbf{x} \quad \mathbf{x}_A], \tag{5.29}$$

where $\mathbf{x}$ and $\mathbf{x}_A$ are given by

$$\mathbf{x} \triangleq [x_1\ x_2\ \ldots\ x_N], \tag{5.30}$$

$$\mathbf{x}_A \triangleq [x_{N+1}\ x_{N+2}\ \ldots\ x_L]. \tag{5.31}$$

Finally, we propose the substitute pmf for $p(\mathbf{x})$ as

$$t_p(\mathbf{x}_E) \triangleq \begin{cases} p(\mathbf{x}), & \text{if } \mathbf{x}_A^T = \mathbf{A}\mathbf{x}^T \\ 0, & \text{otherwise} \end{cases}. \tag{5.32}$$

Clearly, $t_p(\mathbf{x}_E)$ achieves its maximum value at a configuration $\mathbf{x}_{E,MAX}$ which is equal to

$$\mathbf{x}_{E,MAX} \triangleq \arg\max_{\mathbf{x}_E} t_p(\mathbf{x}_E) \tag{5.33}$$

$$= [\mathbf{x}_{MAX} \quad \mathbf{x}_{MAX}\mathbf{A}^T] \tag{5.34}$$

where $\mathbf{x}_{MAX}$ is the configuration maximizing $p(\mathbf{x})$. Due to this property any device which determines the configuration maximizing $t_p(\mathbf{x}_E)$ also determines the configuration maximizing $p(\mathbf{x})$ at the same time.

Now we need to show that $t_p(\mathbf{x}_E)$ can be maximized by an ML codeword decoder. As a first step, we can obtain an equivalent alternative definition of $t_p(\mathbf{x}_E)$ with using parity check constraints as

$$t_p(\mathbf{x}_E) = p(\mathbf{x}) \prod_{i=N+1}^{L} \delta(\mathbf{a}_i\mathbf{x}^T - x_i). \tag{5.35}$$

Inserting the canonical factorization of $p(\mathbf{x})$ into the equation above yields

$$t_p(\mathbf{x}_E) = C_{\mathbb{F}_q^L}\left\{\prod_{i=1}^{L} r_i(\mathbf{a}_i\mathbf{x}^T) \prod_{i=N+1}^{L} \delta(\mathbf{a}_i\mathbf{x}^T - x_i)\right\} \tag{5.36}$$

$$= C_{\mathbb{F}_q^L}\left\{\prod_{i=1}^{N} r_i(\mathbf{a}_i\mathbf{x}^T) \prod_{i=N+1}^{L} r_i(\mathbf{a}_i\mathbf{x}^T)\delta(\mathbf{a}_i\mathbf{x}^T - x_i)\right\}. \tag{5.37}$$

Recall that we assumed the first N $\mathbf{a}_i$ vectors to be of weight one while defining the matrix $\mathbf{A}$. Hence, we may safely assume further that these N $\mathbf{a}_i$ vectors are the canonical basis vectors of $\mathbb{F}_q^N$. With this assumption the factorization of $t_p(\mathbf{x}_E)$ becomes

$$t_p(\mathbf{x}_E) = C_{\mathbb{F}_q^L}\left\{\prod_{i=1}^{N} r_i(x_i) \prod_{i=N+1}^{L} r_i(\mathbf{a}_i\mathbf{x}^T)\delta(\mathbf{a}_i\mathbf{x}^T - x_i)\right\}. \tag{5.38}$$

The only remaining step to obtain a factorization as in (5.22) is to replace $r_i(\mathbf{a}_i\mathbf{x}^T)\delta(\mathbf{a}_i\mathbf{x}^T - x_i)$ with $r_i(x_i)\delta(\mathbf{a}_i\mathbf{x}^T - x_i)$ which yields

$$t_p(\mathbf{x}_E) = C_{\mathbb{F}_q^L}\left\{\prod_{i=1}^{N} r_i(x_i) \prod_{i=N+1}^{L} r_i(x_i)\delta(\mathbf{a}_i\mathbf{x}^T - x_i)\right\} \tag{5.39}$$

$$= C_{\mathbb{F}_q^L}\left\{\prod_{i=1}^{L} r_i(x_i) \prod_{i=N+1}^{L} \delta(\mathbf{a}_i\mathbf{x}^T - x_i)\right\}. \tag{5.40}$$

Since all the factor functions above are either degree one factor functions or parity-check constraints, an ML decoder of a linear code can be utilized to maximize $t_p(\mathbf{x}_E)$.

The parity check matrix of the linear code which can be used to maximize $t_p(\mathbf{x}_E)$ and consequently $p(\mathbf{x})$ at the same time can be found as follows. Let a parity check coefficient vector $\mathbf{h}_i$ of length L be defined as in

$$\mathbf{h}_i \triangleq [\mathbf{a}_{i+N} \quad \mathbf{0}_{1\times(i-1)} \quad -1 \quad \mathbf{0}_{1\times(L-i-N)}] \quad \text{for } 1 \le i \le L-N. \tag{5.41}$$

Equation (5.40) can be expressed using $\mathbf{h}_i$ as

$$t_p(\mathbf{x}_E) = C_{\mathbb{F}_q^L}\left\{\prod_{i=1}^{L} r_i(x_i) \prod_{i=1}^{L-N} \delta(\mathbf{h}_i\mathbf{x}_E^T)\right\}. \tag{5.42}$$

Hence, the parity check matrix $\mathbf{H}$ of the code whose ML codeword decoder can be used to maximize $t_p(\mathbf{x}_E)$ and $p(\mathbf{x})$ is

$$\mathbf{H} \triangleq \begin{bmatrix} \mathbf{h}_1 \\ \mathbf{h}_2 \\ \cdots \\ \mathbf{h}_{L-N} \end{bmatrix} \tag{5.43}$$

$$= [\mathbf{A} \quad -\mathbf{I}_{(L-N)\times(L-N)}]. \tag{5.44}$$

The L input vectors that should be applied to maximize $p(\mathbf{x})$ and $t_p(\mathbf{x}_E)$ are

$$\mathbf{y}_i = \mathcal{L}\{r_i(x)\} \quad \text{for } 1 \le i \le L. \tag{5.45}$$

To sum up, with these input vectors ML codeword decoder of the linear code with parity check matrix $\mathbf{H}$ returns

$$\hat{\mathbf{x}}_{E,ML} = \arg\max_{\mathbf{x}_E} \prod_{i=1}^{L} r_i(x_i) \prod_{i=1}^{L-N} \delta(\mathbf{h}_i\mathbf{x}_E^T) \tag{5.46}$$

$$= \arg\max_{\mathbf{x}_E} t_p(\mathbf{x}_E). \tag{5.47}$$

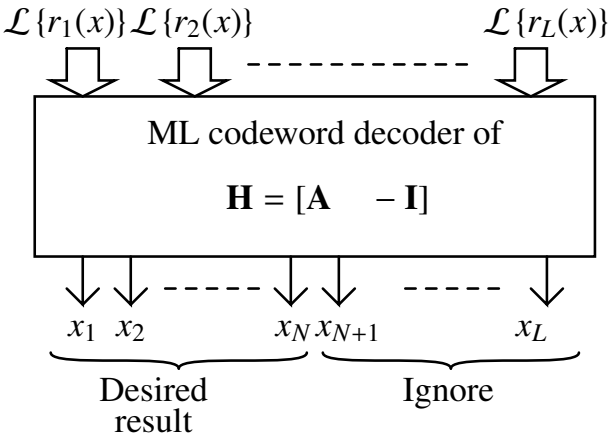

Figure 5.2: Summary of the utilization of an ML codeword decoder for maximizing a pmf

Due to (5.34) the leading N components of $\hat{\mathbf{x}}_{E,ML}$ is the $\mathbf{x}_{MAX}$ vector maximizing $p(\mathbf{x})$ which we are seeking for. We can ignore the rest of the $\hat{\mathbf{x}}_{E,ML}$ vector. The whole process of finding the configuration maximizing $p(\mathbf{x})$ is summarized in Figure 5.2.

It is well known that ML codeword decoding problem is a special instance of maximization of multivariate pmf problems. In this section, we showed that there exists a special ML codeword decoding problem which can handle the maximization task of an arbitrary multivariate pmf. Hence, the reverse of the well known statement above is also true. Therefore, we can conclude that *ML codeword decoding and maximization of multivariate pmfs are equivalent problems*.

5.4 Marginalizing a multivariate pmf by using a symbolwise decoder

Let an $\mathbb{F}_q$-valued random vector $\mathbf{X} = [X_1, X_2, \ldots, X_N]$ be distributed with a $p(\mathbf{x}) \in \mathcal{P}_{\mathbb{F}_q^N}$. The marginal pmf of X_i is

$$\Pr\{X_i = x_i\} = \sum_{\sim\{x_i\}} p(\mathbf{x}). \tag{5.48}$$

A symbolwise decoder can perform this marginalization if $p(\mathbf{x})$ can be factored into degree one factor functions and parity-check constraints, which is not possible for a strictly positive pmf. However, as we did in the previous section, for each $p(\mathbf{x}) \in \mathbb{F}_q$ we can obtain a substitute multivariate pmf which has the desired factorization and can be used in the marginalization of $p(\mathbf{x})$.

We can follow a more straightforward path to obtain this substitute pmf when compared to the previous section. Inserting the canonical factorization of $p(\mathbf{x})$ into the marginalization sum above yields

$$\begin{aligned}
\Pr\{X_i = x_i\} &= \sum_{\sim\{x_i\}} C_{\mathbb{F}_q^N}\left\{\prod_{i=1}^{L} r_i(\mathbf{a}_i\mathbf{x}^T)\right\} && (5.49)\\
&= C_{\mathbb{F}_q}\left\{\sum_{\sim\{x_i\}}\prod_{i=1}^{L} r_i(\mathbf{a}_i\mathbf{x}^T)\right\} && (5.50)\\
&= C_{\mathbb{F}_q}\left\{\sum_{\sim\{x_i\}}\prod_{i=1}^{N} r_i(x_i)\prod_{i=N+1}^{L} r_i(\mathbf{a}_i\mathbf{x}^T)\right\}, && (5.51)
\end{aligned}$$

where we make the same assumptions as in the previous section about the canonical factorization of $p(\mathbf{x})$. This equation shows that N of the factor functions are already of degree one. The remaining factor functions, which are SPC constraints, can be expressed by using the sifting property of the Kronecker delta function as

$$r_i(\mathbf{a}_i\mathbf{x}^T) = \sum_{\forall x_i\in\mathbb{F}_q} \delta(\mathbf{a}_i\mathbf{x}^T - x_i)r_i(x_i) \quad \text{for } N+1 \le i \le L. \tag{5.52}$$

Since i is greater than N, x_i above is not a component of vector $\mathbf{x}$ and is just a dummy variable. Using this identity in the marginalization sum gives

$$\begin{aligned}
\Pr\{X_i = x_i\} &= C_{\mathbb{F}_q}\left\{\sum_{\sim\{x_i\}}\prod_{i=1}^{N} r_i(x_i)\prod_{i=N+1}^{L}\sum_{\forall x_i\in\mathbb{F}_q}\delta(\mathbf{a}_i\mathbf{x}^T - x_i)r_i(x_i)\right\} && (5.53)\\
&= C_{\mathbb{F}_q}\left\{\sum_{\sim\{x_i\}}\sum_{\forall\mathbf{x}_A\in\mathbb{F}_q^{L-N}}\prod_{i=1}^{L} r_i(x_i)\prod_{i=N+1}^{L}\delta(\mathbf{a}_i\mathbf{x}^T - x_i)\right\}, && (5.54)
\end{aligned}$$

where $\mathbf{x}_A$ is as defined in (5.31). Thanks to the summary notation the summation running over $\mathbf{x}_A$ can be merged to the first summation which yields

$$\Pr\{X_i = x_i\} = C_{\mathbb{F}_q}\left\{\sum_{\sim\{x_i\}}\prod_{i=1}^{L} r_i(x_i)\prod_{i=1}^{L-N}\delta(\mathbf{h}_i\mathbf{x}_E^T)\right\}, \tag{5.55}$$

where $\mathbf{x}_E$ and $\mathbf{h}_i$ are defined in (5.29) and (5.41) respectively. Notice that the two products above is the factorization of $t_p(\mathbf{x}_E)$, which is defined in (5.32), given in (5.42). Therefore,

$$\begin{aligned}
\Pr\{X_i = x_i\} &= C_{\mathbb{F}_q}\left\{\sum_{\sim\{x_i\}} t_p(\mathbf{x}_E)\right\} && (5.56)\\
&= \sum_{\sim\{x_i\}} t_p(\mathbf{x}_E). && (5.57)
\end{aligned}$$

This result shows that the marginal probability of X_i, $\Pr\{X_i = x_i\}$, can be computed either by marginalizing $p(\mathbf{x})$ or by marginalizing $t_p(\mathbf{x}_E)$.

Similar to the maximization of $t_p(\mathbf{x}_E)$, marginalization of $t_p(\mathbf{x}_E)$ can be accomplished by the *symbolwise decoder* of the linear code with parity check $\mathbf{H}$ defined in (5.43). When input vectors $\mathbf{y}_i = \mathcal{L}\{r_i(x)\}$ is applied to this symbolwise decoder it returns the marginal probabilities associated with the APP

$$\Pr\{\mathbf{X}_E = \mathbf{x}_E | \mathbf{Y} = \mathbf{y}\} = C_{\mathbb{F}_q^L} \left\{ \prod_{i=1}^{L} r_i(x_i) \prod_{i=1}^{L-N} \delta(\mathbf{h}_i \mathbf{x}_E^T) \right\}, \tag{5.58}$$

which is equal to $t_p(\mathbf{x}_E)$. Hence, this decoder is capable of both marginalizing $t_p(\mathbf{x}_E)$ and consequently $p(\mathbf{x})$ at the same time.

We could achieve the result given in (5.57) through a much shorter path if we started from the definition of $t_p(\mathbf{x}_E)$ given in (5.32). We preferred the path followed above to this shorter path, since the path above explains how we reached to the proposed definition of $t_p(\mathbf{x}_E)$ which is the most critical part of the previous section.

It is very well known that symbolwise decoding is an instance of marginalization problems in general. In this section we showed that marginalization of multivariate pmfs can be expressed as a particular symbolwise decoding problem. Hence, it can be concluded that *symbolwise decoding and marginalization are equivalent problems*.

5.5 The decoder of the dual Hamming code as the universal inference machine

In the previous two sections we have shown that the ML codeword and symbolwise decoders of the linear code with parity check matrix $\mathbf{H}$ can be used to maximize and marginalize any pmf in $\mathcal{P}_{\mathbb{F}_q^N}$. This parity check matrix belongs to the dual code of a very well known code from coding theory. Recall that the matrix $\mathbf{H}$ is as defined as

$$\mathbf{H} = \begin{bmatrix} \mathbf{A} & -\mathbf{I}_{(L-N)\times(L-N)} \end{bmatrix} \tag{5.59}$$

$$= \begin{bmatrix} \mathbf{a}_{N+1} & -1 & 0 & \cdots & 0 \\ \mathbf{a}_{N+2} & 0 & -1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & 0 \\ \mathbf{a}_L & 0 & 0 & \cdots & -1 \end{bmatrix}. \tag{5.60}$$

The generator matrix of this code is

$$\mathbf{G} = \begin{bmatrix} \mathbf{I}_{N\times N} & \mathbf{A}^T \end{bmatrix}. \tag{5.61}$$

In Section 5.3 we assumed that the first N $\mathbf{a}_i$ vectors are the canonical basis vectors of $\mathbb{F}_q^N$. Therefore, the generator matrix can be written as

$$\mathbf{G} = \begin{bmatrix} \mathbf{a}_1^T & \mathbf{a}_2^T & \cdots & \mathbf{a}_L^T \end{bmatrix}. \tag{5.62}$$

Recall that all $\mathbf{a}_i$ vectors were pairwise linearly independent and L was equal to $\frac{q^N-1}{q-1}$. Therefore, the generator matrix $\mathbf{G}$ given above is actually the parity check matrix of the Hamming code in $\mathbb{F}_q$ of length L. Hence, the parity check matrix $\mathbf{H}$ given in (5.59) is the parity check matrix of the dual Hamming code in $\mathbb{F}_q$ of length L. Consequently, the ML codeword decoder of the (L, N) dual Hamming code can be configured by adjusting its inputs to maximize any pmf in $\mathcal{P}_{\mathbb{F}_q^N}$. Similarly, the symbolwise decoder of the (L, N) dual Hamming code can be used as an apparatus to marginalize any pmf in $\mathcal{P}_{\mathbb{F}_q^N}$. Therefore, the decoders of the dual Hamming codes are universal inference machines.

5.6 Performing inference on special pmfs by decoders

In the previous sections we have shown that the decoders of the (L, N) dual Hamming code designed for the channel model given in (5.2) can be used to perform inference on any pmf in $\mathcal{P}_{\mathbb{F}_q^N}$. The analysis presented in the previous sections is for the most general case. Decoders of shorter codes designed for simpler channel models can be employed to perform inference on some pmfs enjoying special properties in their canonical factorization.

5.6.1 Performing inference with the decoders of shorter codes

In Chapter 4 we investigated the canonical factorizations of some special pmfs. The canonical factorization of these special pmfs consisted of less than $\frac{q^N-1}{q-1}$ SPC factors. We can perform inference on these special pmfs by using the decoders of the codes whose parity check matrices are the sub-matrices of the (L, N) dual Hamming code.

Suppose that we would like to perform inference on a special pmf $p(\mathbf{x}) \in \mathcal{P}_{\mathbb{F}_q^N}$ whose canonical

factorization can be expressed as

$$p(\mathbf{x}) = C_{\mathbb{F}_q^N} \left\{ \prod_{\mathbf{a}_i \in \mathcal{D}} r_i(\mathbf{a}_i \mathbf{x}^T) \right\}, \tag{5.63}$$

where $\mathcal{D}$ is a subset of $\mathcal{H}$ and $r_i(\mathbf{a}_i \mathbf{x}^T)$ is the projection of $p(\mathbf{x})$ onto $\mathrm{im}\{\mathcal{S}_{\mathbf{a}_i}\}$. Let $\mathcal{B}$ be a subset of $\mathcal{D}$ which consists of all of the parity check coefficient vectors in $\mathcal{D}$ of *weight two or more*. Moreover, let $\mathbf{B}$ be a $|\mathcal{B}| \times N$ matrix whose rows are the vectors in $\mathcal{B}$. Then we may define the substitute pmf $t_p(\mathbf{x}_F)$ which can be used to perform inference on $p(\mathbf{x})$ as

$$t_p(\mathbf{x}_F) \triangleq \begin{cases} p(\mathbf{x}), & \text{if } \mathbf{x}_B^T = \mathbf{B}\mathbf{x}^T \\ 0, & \text{otherwise} \end{cases}. \tag{5.64}$$

where $\mathbf{x}_B$ and $\mathbf{x}_F$ are given by

$$\mathbf{x}_B \triangleq [x_{N+1} \; x_{N+2} \; \ldots \; x_{|\mathcal{B}|+N}], \tag{5.65}$$

$$\mathbf{x}_F \triangleq [\mathbf{x} \quad \mathbf{x}_A]. \tag{5.66}$$

It can be shown through a similar path to the one in Section 5.3 and Section 5.4 that we can maximize or marginalize $t_p(\mathbf{x}_F)$ if we wish to determine the configuration maximizing $p(\mathbf{x})$ or marginalize $p(\mathbf{x})$. Moreover, we can use the ML codeword and symbolwise decoders of the linear code with parity check matrix

$$\mathbf{H}_S \triangleq [\mathbf{B} \quad -\mathbf{I}] \tag{5.67}$$

to maximize or marginalize $t_p(\mathbf{x}_F)$. Hence, these decoders can be used to maximize or marginalize $p(\mathbf{x})$.

As in Section 5.3 and Section 5.4, the ML codeword and symbolwise decoders of the linear code described by parity check matrix $\mathbf{H}_S$ should be configured to perform inference on $p(\mathbf{x})$ by applying a certain set of inputs. The i^{th} of these inputs is $\mathcal{L}^+ \{v_i(x)\}(x)$ where $v_i(\mathbf{b}_i \mathbf{x}^T)$ is the projection of $p(\mathbf{x})$ onto $\mathrm{im}\{\mathcal{S}_{\mathbf{b}_i}\}$, and $\mathbf{b}_i$ is the i^{th} canonical basis vector of $\mathbb{F}_q^N$ if i is less than or equal to N and $(i-N)^{th}$ row of $\mathbf{B}$ otherwise.

Since $\mathcal{D}$ is a subset of $\mathcal{H}$, $\mathbf{B}$ is a sub-matrix of $\mathbf{A}$ defined in (5.28). Consequently, $\mathbf{H}_S$ is a sub-matrix of $\mathbf{H}$ defined in (5.43). Therefore, implementing the decoder associated with $\mathbf{H}_S$ is easier than implementing the decoder associated with $\mathbf{H}$.

Actually, there are many linear codes whose decoders can be employed to perform inference on $t_p(\mathbf{x}_F)$ and $p(\mathbf{x})$ at the same time. For instance the decoders of the linear codes with parity

check matrices in the form given below can be used in performing inference on $p(\mathbf{x})$,

$$\mathbf{H}_{SE} \triangleq [\mathbf{C} \ -\mathbf{I}], \tag{5.68}$$

where $\mathbf{C}$ is a sub-matrix of $\mathbf{A}$ such that it contains all rows of $\mathbf{B}$ and some more. The linear code with parity check matrix $\mathbf{H}_S$ is the one with the shortest length among these codes. At a first glance preferring the decoder of a longer code to a shorter one might seem useless while solving the same inference problem. However, in the next chapter we are going to provide some examples in which choosing the decoder of the longer code might be advantageous.

If a linear code has a parity check matrix which can be obtained by permuting the columns of $\mathbf{H}_{SE}$ then the decoders of this code can also be employed in maximizing or marginalizing $p(\mathbf{x})$. However, in order to obtain the desired result we need to apply permuted inputs.

5.6.2 Performing inference by decoders designed for simpler channels

Let $\mathcal{Y}$ be the output alphabet of a communication channel which might be a finite set, real field, complex field, or a vector space. If there exists a sequence of channel outputs $y_1, y_2, \ldots, y_{|\mathcal{H}|}$ in $\mathcal{Y}$ such that a multivariate pmf $p(\mathbf{x}) \in \mathcal{P}_{\mathbb{F}_q^N}$ can be expressed as

$$p(\mathbf{x}) = C_{\mathbb{F}_q^N} \left\{ \prod_{\mathbf{a}_i \in \mathcal{H}} \Pr\{Y = y_i | X = \mathbf{a}_i \mathbf{x}^T\} \right\}, \tag{5.69}$$

where $\Pr\{Y = y|X = x\}$ denotes the likelihood function of the channel, then the decoder of the dual Hamming code designed for this channel can be employed to perform inference on $p(\mathbf{x})$. The inputs that should be applied to this decoder to perform inference on $p(\mathbf{x})$ are obviously $y_1, y_2, \ldots, y_{|\mathcal{H}|}$.

In some problems preferring other channel models to the one described in (5.2) might be simpler. We selected the channel model therein since for each $r(x) \in \mathbb{F}_q$ there exists a $\mathbf{y} \in \mathbb{R}^q$ such that $r(x) = C_{\mathbb{F}_q}\{\Pr\{\mathbf{Y} = \mathbf{y}|X = x\}\}$. Consequently, the decoders of the dual Hamming code designed for this channel can be employed to perform inference on any pmf $p(\mathbf{x}) \in \mathbb{F}_q^N$.

5.7 The Generic Factor Graph and Equivalent Tanner graph

In Chapter 3, it is shown that the canonical factorization of any multivariate pmf $p(\mathbf{x})$ in $\mathcal{P}_{\mathbb{F}_q^N}$ exists which is given by

$$p(\mathbf{x}) = C_{\mathbb{F}_q^N} \left\{ \prod_{i=1}^{L} r_i(\mathbf{a}_i \mathbf{x}^T) \right\}. \tag{5.70}$$

In this chapter, we made some assumptions on $\mathbf{a}_i$. We assumed without loss of generality that the first N parity check coefficient vectors, $\mathbf{a}_1$, $\mathbf{a}_2$, ..., $\mathbf{a}_N$, are the canonical basis vectors of $\mathbb{F}_q^N$. Therefore, the canonical factorization of any $p(\mathbf{x})$ becomes

$$p(\mathbf{x}) = C_{\mathbb{F}_q^N} \left\{ \prod_{i=1}^{N} r_i(x_i) \prod_{i=N+1}^{L} r_i(\mathbf{a}_i \mathbf{x}^T) \right\}. \tag{5.71}$$

The factor graph representing this factorization is shown in Figure 5.3-a. This factor graph can represent any pmf in $\mathcal{P}_{\mathbb{F}_q^N}$ since all of the pmfs in $\mathcal{P}_{\mathbb{F}_q^N}$ has a factorization given above. The only difference between any two factor graphs representing two different joint pmfs are the factor functions in the factor graph.

In this chapter, we showed that performing inference on $p(\mathbf{x})$ is equivalent to performing inference on $t_p(\mathbf{x}_E)$ which is defined in (5.32). The factorization of $t_p(\mathbf{x}_E)$ given in 5.40 is represented by the Tanner graph shown in Figure 5.3-b.in Hence, this Tanner graph is the equivalent Tanner graph representing the canonical factorization. While transforming the factor graph in Figure 5.3-a to the Tanner graph in in Figure 5.3-b, auxiliary variable nodes representing the variables x_{N+1}, x_{N+2}, ..., x_L are added. These auxiliary variables are very different from the hidden state nodes introduced in the Wiberg style Tanner graphs [6].

5.8 Importance

Using channel decoders for inference tasks beyond decoding is important mainly in two aspects. Firstly, using a channel decoder for an inference task provides new hardware options in the solution of the inference problems. Among these hardware options the analog probability propagation technique is important in particular [14]. Secondly, new approximate algorithms for the solution of the inference problems can be developed by using the sub-optimal decoders of the codes, which have been studied for a long time.

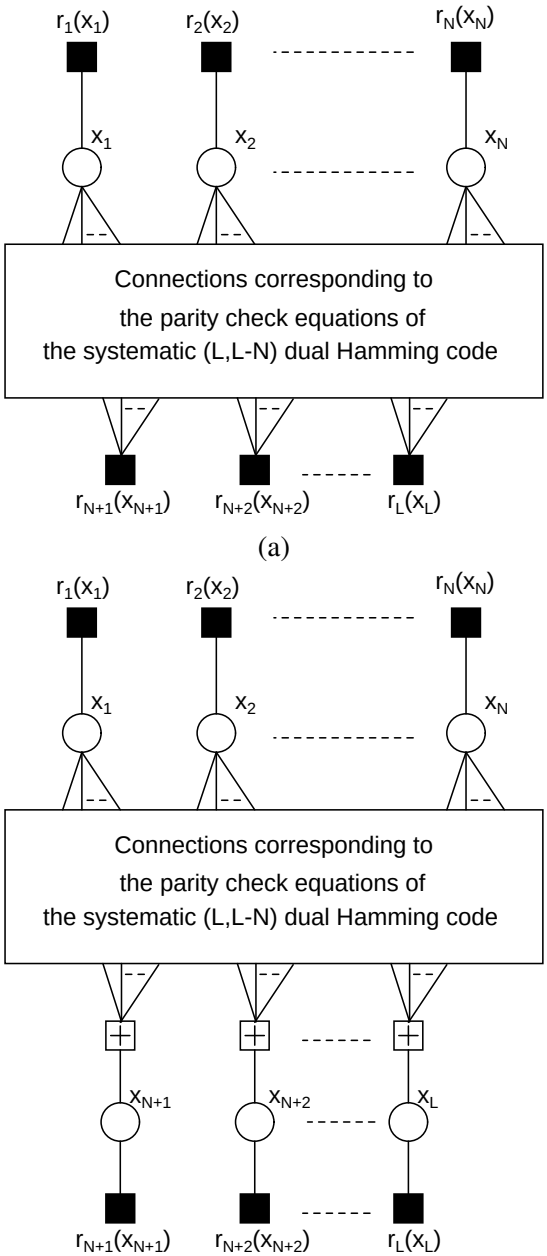

Figure 5.3: (a) The generic factor graph which can represent any $p(\mathbf{x})$ in $\mathcal{P}_{\mathbb{F}_q^N}$. (b) The equivalent Tanner graph of the generic Tanner graph.

5.8.1 Performing inference with probability propagation in analog VLSI

The semiconductor devices such as transistors and diodes are the most primitive building blocks of any electronic device today. By their very nature these devices are *nonlinear*. Over the last few decades engineers developed ways to cope with this nonlinearity. While designing analog circuitry engineers restrict the operation of the circuit to such a region in which these devices behave almost linearly. Another way to cope with nonlinearity of these devices is avoiding analog circuits as much as possible and trying to implement everything in digital. The signals flowing in a digital circuit are so large that transistors behave like switches. Hence, digital circuits are robust against the nonlinearity of the transistors. Digital circuits are also robust against other factors such as component mismatch and noise. Due to these and some other advantages digital circuits are usually preferred to analog circuits.

However, Carver Mead, who is one of the pioneers of the VLSI revolution, claimed in his book [34] that digital computation is inefficient and analog computation is the way to achieve the capacity and the efficiency of the brains of the animals. Moreover, he claimed that analog computation can be made as robust as digital computation to the factors such as noise and component mismatch. He provided many practical examples to support his claims in his book.

A decade after Mead's book, another evidence arise from coding theory to support his claims. Just two operations are sufficient to perform soft-input soft-output decoding. These operations are addition, which can be easily implemented with analog circuitry, and the hyperbolic tangent function [1]. Since the differential pair exhibits tangent hyperbolic function this second function can also be implemented with analog circuits. Motivated with this idea, Loeliger and his group designed and tested analog circuits to perform decoding of channel codes [15, 14]. They report that their analog decoding circuitry consumes two orders of magnitude less power than their digital counterparts. This efficiency arises from the fact that their analog decoding circuit does not fight with the nonlinearities of the transistors but exploits those nonlinearities [15]. They also report that these circuits are robust to component mismatch.

Loeliger's "probability propagation in analog VLSI" has an important limitation. This approach can be applied to probabilistic inference problems if a condition related to the factorization of the multivariate pmf under concern is satisfied. This condition states that the pmf

should be able to be expressed as a product of zero-one valued functions and functions of degree one [14]. Although this condition is satisfied in decoding problems, it is not satisfied in other problems arising in communication theory such as channel equalization and MIMO detection. Hence, equalizers or MIMO detectors could not be built directly with their brilliant idea whereas decoders could. A pure decoder implemented with probability propagation in analog is not very useful without implementing the equalizer or detector in analog since the interface required between the decoder and the equalizer (or detector) would cancel all the efficiency of the analog decoder.

In this chapter, we showed that inference problems can be solved by using channel decoders. Hence, it is possible to solve the equalization or MIMO detection problems by decoders. Consequently, the results presented in this chapter, allows us to implement channel equalizers or MIMO detectors with the very efficient analog probability propagation approach. It is reasonable to expect, based on the experience on analog decoding, that such receiver blocks would be two orders of magnitude smaller in size and consumes two orders of magnitude less power than current receivers. Probably this aspect will be the most important contribution of this thesis.

5.8.2 New approximate inference algorithms

The iterative sum-product algorithm running on loopy Tanner graphs is proven to be efficient decoding algorithm for various codes. The sum-product algorithm is characterized by the Tanner graph representing the code. A code might be represented with many different parity check matrices. For each parity check matrix, more than one Tanner graphs might be obtained representing the code. Hence, for each code we have various alternative Tanner graphs to represent the code. Consequently, we may have various versions of the sum-product algorithm to decode the same code. Each of these alternative versions have different characteristics in terms of complexity and performance [7]. Therefore, employing a channel decoder to perform an inference task allows us to choose among different sum-product algorithm versions to handle the inference task. Hence, new approximate inference algorithms can be developed in this manner. We provide an example on MIMO detection in the next chapter.

CHAPTER 6

USING CHANNEL DECODERS AS DETECTORS

6.1 Introduction

This chapter contains examples to the idea presented in Chapter 5 by showing how to employ channel decoders as the detectors of communication receivers. One of these examples which is MIMO detection by using the decoder of a tail biting convolutional code demonstrates that new inference algorithms with low complexity can be developed by employing channel decoders for other purposes.

Unfortunately, some of the derivations presented in this chapter might appear quite tedious, Sections 6.3, 6.4, and 6.5 in particular. Actually, the derivations in these sections are straightforward applications of the methods presented in the previous chapter. Most of these derivations are so straight forward that they can be derived with symbolic programming. Indeed, we used the GiNaC symbolic programming library in C++ while deriving some of the cumbersome derivations presented in this chapter. Hence, reporting and following these derivations is much more difficult than deriving them. However, these sections include examples to make the subject more concrete. These examples also demonstrate how the same decoder can be used for different purposes by changing its inputs.

This chapter begins with analyzing the multiple-input single-output (MISO) detection. Then the results obtained in Section 6.2 are used to derive the channel decoder which can be used in the detection of naturally mapped pulse amplitude modulation (PAM) signals in Section 6.3. Section 6.4 explains the detection of gray mapped PAM signals by using channel decoders. Section 6.5 investigates the multiple-input multiple-output (MIMO) detection of QPSK signal by using decoders. Special attention is paid to the MIMO detection of QPSK signals by using

the decoders of tail biting convolutional codes in Section 6.6. This section also includes some simulation results. This chapter ends with briefly reporting that the Viterbi and BCJR decoders of the convolutional codes can be used channel equalizers.

6.2 MISO detection of q-ary PSK signaling with prime q by using a channel decoder

The MISO detection of q-ary PSK signaling under additive Gaussian noise is the simplest task (in terms of derivation) to be handled by a decoder. Moreover, analyzing this case first helps to transform other detection problems to decoding problems. ML MISO detection task is finding the most likely input sequence given the received symbol. This task can be handled by ML codeword decoders. Soft output MISO detection is the computation of marginal a posteriori probabilities. This task can be handled by symbolwise decoders.

6.2.1 Signal Model

Let $\mu_q(x)$ be a function from $\mathbb{F}_q$ to $\mathbb{C}$ representing the q-ary PSK [1] mapping, i.e.

$$\mu_q(x) \triangleq \exp\left(j\frac{2\pi}{q}\text{int}(x)\right), \tag{6.1}$$

where int(.) denotes the usual mapping from $\mathbb{F}_q$ to $\mathbb{N}$. Let a complex-valued random variable Y be related to an $\mathbb{F}_q$-valued random vector $\mathbf{X} = [X_1, X_2, \ldots, X_N]$ as

$$Y \triangleq \sum_{i=1}^{N} h_i \mu_q(X_i) + Z \tag{6.2}$$

where h_i is a complex constant and Z is a zero mean circularly symmetric complex Gaussian noise with $\mathbf{E}[ZZ^*] = 2\sigma^2$. Clearly, Y models the received symbol after the symbols X_1, X_2, ..., X_N are modulated with q-ary PSK and passed through a $1 \times N$ multi-input single output (MISO) channel with channel coefficients h_i. With these assumptions the a posteriori pmf $\mathbf{X}$ is

$$\Pr\{\mathbf{X} = \mathbf{x}|Y = y\} = C_{\mathbb{F}_q^N}\left\{\exp\left(-\frac{\left|y - \sum_{i=1}^{N} h_i\mu_q(x_i)\right|^2}{2\sigma^2}\right)\right\}, \tag{6.3}$$

where $\mathbf{x} = [x_1, x_2, \ldots, x_N]$. We assume perfect channel information is known at the receiver side.

[1] q-ary PSK is not the same as QPSK.

6.2.2 The canonical factorization of the joint a posteriori pmf

The first step in determining the linear code whose ML codeword (symbolwise) decoder can be used to maximize (marginalize) the a posteriori pmf $\Pr\{\mathbf{X} = \mathbf{x}|Y = y\}$ is obtaining the canonical factorization of the a posteriori pmf of $\mathbf{X}$. The generic procedure of obtaining this canonical factorization is explained in detail in Chapter 3, which could have been prohibitively tedious for this problem. Fortunately, the joint a posteriori pmf in this problem, $\Pr\{\mathbf{X} = \mathbf{x}|Y = y\}$, enjoys many special properties so that deriving its canonical factorization is easier.

Let $p(\mathbf{x})$ denote $\Pr\{\mathbf{X} = \mathbf{x}|Y = y\}$. As shown in Appendix A.4.1, $p(\mathbf{x})$ can be factored as in

$$p(\mathbf{x}) = C_{\mathbb{F}_q^N} \left\{ \prod_{i=1}^{N} \exp\left(\frac{2\mathrm{Re}\left\{ y h_i^* \mu_q(x_i)^* \right\}}{2\sigma^2} \right) \prod_{j=2}^{N} \prod_{i=1}^{j-1} \exp\left(-\frac{2\mathrm{Re}\left\{ h_i h_j^* \mu_q(x_i) \mu_q\left(x_j\right)^* \right\}}{2\sigma^2} \right) \right\}. \quad (6.4)$$

Since we have a known factorization for $p(\mathbf{x})$, we can apply Theorem 4.4 to obtain the *canonical factorization* of $p(\mathbf{x})$ as explained below.

Let two functions $\gamma(\omega; \rho, \sigma)$ and $\theta(\omega_1, \omega_2; \chi, \sigma)$ be defined as in

$$\gamma(\omega; \rho, \sigma) \triangleq \exp\left(\frac{2\mathrm{Re}\left\{ \rho \mu_q(\omega)^* \right\}}{2\sigma^2} \right), \quad (6.5)$$

$$\theta(\omega_1, \omega_2; \chi, \sigma) \triangleq \exp\left(-\frac{2\mathrm{Re}\left\{ \chi \mu_q(\omega_1) \mu_q(\omega_2)^* \right\}}{2\sigma^2} \right), \quad (6.6)$$

for ω, ω_1, ω_2 in $\mathbb{F}_q$, σ in $\mathbb{R}$, and ρ, χ in $\mathbb{C}$. The function $\gamma(\omega; \rho, \sigma)$ is nothing but the likelihood function of ω when it is modulated with q-ary PSK, passed through an additive white Gaussian noise (AWGN) channel with power spectral density (PSD) $N_0/2 = \sigma^2$, and given that the value at the output of the matched filter is ρ. Using these functions the factorization of $p(\mathbf{x})$ becomes

$$p(\mathbf{x}) = C_{\mathbb{F}_q^N} \left\{ \prod_{i=1}^{N} \gamma(x_i; y h_i^*, \sigma) \prod_{j=2}^{N} \prod_{i=1}^{j-1} \theta(x_i, x_j; h_i h_j^*, \sigma) \right\}. \quad (6.7)$$

Notice that the factorization of $p(\mathbf{x})$ given above is composed of degree one and degree two factors only [2]. Therefore, the *canonical factorization* of $p(\mathbf{x})$ should be composed of SPC factors of degree one and two due to Theorem 4.4. The SPC factors of degree one composing $p(\mathbf{x})$ are simply the normalizations of $\gamma(x_i; y h_i^*, \sigma)$'s.

The SPC factors of degree two composing $p(\mathbf{x})$ can be derived by obtaining the canonical factorization of $\theta(x_i, x_j; h_i h_j^*, \sigma)$. The straightforward way of deriving the canonical factoriza-

[2] We regard ρ, χ and σ as parameters of functions $\gamma(.;.)$ and $\theta(.;.)$, not their arguments.

tion of $\theta(x_i, x_j; h_i h_j^*, \sigma)$ might be projecting this function onto the subspaces $\text{im}\left\{\mathcal{S}_{(\mathbf{f}_i+\alpha\mathbf{f}_j)}\right\}$ for all nonzero $\alpha \in \mathbb{F}_q$, where $\mathbf{f}_i$ is the i^{th} canonical basis vector of $\mathbb{F}_q^N$. However, the required canonical factorization can be obtained in a simpler way by exploiting the fact that q is assumed to be a prime number in this section. Since q is a prime number, $\mathbb{F}_q$ is a prime field. Consequently, the subtraction in $\mathbb{F}_q$ is the subtraction modulo q. Due to this fact,

$$\mu_q(\omega_1)\,\mu_q(\omega_2)^* = \mu_q(\omega_1 - \omega_2). \tag{6.8}$$

Therefore,

$$\begin{aligned}\theta(\omega_i, \omega_j; \chi, \sigma) &= \exp\left(-\frac{2\text{Re}\left\{\chi\mu_q(\omega_1 - \omega_2)\right\}}{2\sigma^2}\right) &(6.9)\\ &= \gamma(\omega_1 - \omega_2; -\chi, \sigma). &(6.10)\end{aligned}$$

Inserting this result into (6.7) yields,

$$p(\mathbf{x}) = C_{\mathbb{F}_q^N}\left\{\prod_{i=1}^{N}\gamma(x_i; yh_i^*, \sigma)\prod_{j=2}^{N}\prod_{i=1}^{j-1}\gamma(x_i - x_j; -h_i h_j^*, \sigma)\right\}. \tag{6.11}$$

We can define pmfs in $\mathcal{P}_{\mathbb{F}_q}$ by scaling $\gamma(x; yh_i^*, \sigma)$ and $\gamma(x; -h_i h_j^*, \sigma)$ as in

$$\begin{aligned}r_i(x) &\triangleq C_{\mathbb{F}_q}\left\{\gamma(x; yh_i^*, \sigma)\right\}, &(6.12)\\ r_{i,j}(x) &\triangleq C_{\mathbb{F}_q}\left\{\gamma(x; -h_i h_j^*, \sigma)\right\}. &(6.13)\end{aligned}$$

The factorization of $p(\mathbf{x})$ can be expressed by using these pmfs as

$$\begin{aligned}p(\mathbf{x}) &= C_{\mathbb{F}_q^N}\left\{\prod_{i=1}^{N} r_i(x_i)\prod_{j=2}^{N}\prod_{i=1}^{j-1} r_{i,j}(x_i - x_j)\right\} &(6.14)\\ &= C_{\mathbb{F}_q^N}\left\{\prod_{i=1}^{N} r_i(\mathbf{f}_i\mathbf{x}^T)\prod_{j=2}^{N}\prod_{i=1}^{j-1} r_{i,j}(\mathbf{a}_{i,j}\mathbf{x}^T)\right\}, &(6.15)\end{aligned}$$

where $\mathbf{a}_{i,j}$ is

$$\mathbf{a}_{i,j} \triangleq \mathbf{f}_i - \mathbf{f}_j. \tag{6.16}$$

Notice that all the factor functions in factorization above are SPC factors. Moreover, the parity check coefficient vectors of all SPC factors are pairwise linearly independent. Hence, due to Definition 4, the factorization of $p(\mathbf{x})$ given in (6.15) is the canonical factorization of $p(\mathbf{x})$.

6.2.3 The decoders which are able to perform inference on the joint a posteriori pmf

As explained in Section 5.5 the ML codeword and symbolwise decoders of the dual Hamming code of length $\frac{q^N-1}{q-1}$ can perform inference on $p(\mathbf{x})$. However, since the canonical factorization

of $p(\mathbf{x})$ given in (6.15) consists of less than $\frac{q^N-1}{q-1}$ SPC factors, the ML codeword or symbol-wise decoders of a shorter code can be employed for maximizing and marginalizing $p(\mathbf{x})$ as discussed in Section 5.6. Following the discussion in Section 5.6 the parity check matrix of this code whose decoder can be employed in the demodulation of $1 \times N$ MISO system is

$$\mathbf{H}_{qPSK}(N) \triangleq \begin{bmatrix} \mathbf{a}_{1,2} & \\ \mathbf{a}_{1,3} & \\ \mathbf{a}_{2,3} & \\ \vdots & \\ \mathbf{a}_{1,N} & -\mathbf{I}_{\frac{N(N-1)}{2}\times\frac{N(N-1)}{2}} \\ \mathbf{a}_{2,N} & \\ \vdots & \\ \mathbf{a}_{N-1,N} & \end{bmatrix}. \tag{6.17}$$

For a neater representation of $\mathbf{H}_{qPSK}(N)$, we define a matrix parameterized on i and N $\mathbf{K}(i,N)$ as

$$\mathbf{K}(i,N) \triangleq \begin{bmatrix} \mathbf{I}_{i\times i} & -\mathbf{1}_{i\times 1} & \mathbf{0}_{i\times(N-i-1)} \end{bmatrix}. \tag{6.18}$$

Then $\mathbf{H}_{qPSK}(N)$ can be expressed as

$$\mathbf{H}_{qPSK}(N) = \begin{bmatrix} \mathbf{K}(1,N) & \\ \mathbf{K}(2,N) & \\ \vdots & -\mathbf{I}_{\frac{N(N-1)}{2}\times\frac{N(N-1)}{2}} \\ \mathbf{K}(N,N) & \end{bmatrix}. \tag{6.19}$$

The complete specification of a decoder of a linear code consists of a parity check matrix and a channel model. The parity check matrix of the decoders which can detect received symbols of $1 \times N$ MISO system are explained above. As the channel model we can use the one described in (5.2). However, we can use a more natural channel model in this case as explained in Section 5.6.2. Recall that the factorization of $p(\mathbf{x})$ given in (6.11) is composed of likelihood functions of the channel which first modulates an $\mathbb{F}_q$-valued symbol with q-ary PSK and then passes through an AWGN channel with PSD $N_0/2 = \sigma^2$. Therefore, the received symbols of $1 \times N$ MISO system can be detected with the decoders of the code with parity check matrix $\mathbf{H}_{qPSK}$ which is designed for q-ary PSK modulation and AWGN channel with variance σ^2. In order to achieve the desired detection inputs that should be applied to these decoders are

components of the vector given below.

$$[yh_1^* \; yh_2^* \; \ldots \; yh_N^* \; - h_1h_2^* \; - h_1h_3^* \; - h_2h_3^* \; \ldots \; - h_1h_N^* \; - h_2h_N^* \; \ldots - h_{N-1}h_N^*]$$

We can also use a modification of the same decoder which is designed for standard noise with $\sigma^2 = 1$. In this case all of the inputs given above should be scaled by $\frac{1}{\sigma}$.

Example 6.1 *This example demonstrates how can we employ a symbolwise decoder to compute the marginal APPs in a* 1×4 *MISO system. Let a complex-valued random variable Y be given as*

$$Y = \sum_{i=1}^{4} h_i \mu_q (X_i) + Z \tag{6.20}$$

where X_i *is an* $\mathbb{F}_q$*-valued random variable and Z is the circularly symmetric Gaussian noise with* $\mathbf{E}[ZZ^*] = 2$*. Our aim is to compute* $\Pr\{X_i = x_i | Y = y\}$ *by using a symbolwise decoder. As explained above the parity check matrix of this decoder is*

$$\mathbf{H}_{qPSK}(4) = \begin{bmatrix} \mathbf{K}(1,4) & \\ \mathbf{K}(2,4) & -\mathbf{I}_{6\times 6} \\ \mathbf{K}(3,4) & \end{bmatrix} \tag{6.21}$$

$$= \begin{bmatrix} 1 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 1 & 0 & 0 & -1 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix}. \tag{6.22}$$

The input vector that should be applied to this decoder is

$$[yh_1^* \quad yh_2^* \quad yh_3^* \quad yh_4^* \quad - h_1h_2^* \quad - h_1h_3^* \quad - h_2h_3^* \quad - h_1h_4^* \quad - h_2h_4^* \quad - h_3h_4^*].$$

Notice that configuring the demodulator for a new observation and new set of channel coefficients requires only changing the inputs to the decoder. This example is illustrated in Figure 6.1.

6.3 Channel decoders as detectors of naturally mapped M-PAM

In this section we show how to demodulate the naturally mapped M-PAM modulation by using a channel decoder. Let $\eta_N(\mathbf{x})$ be a function from $\mathbb{F}_2^N$ to $\mathbb{R}$ which maps binary valued vectors

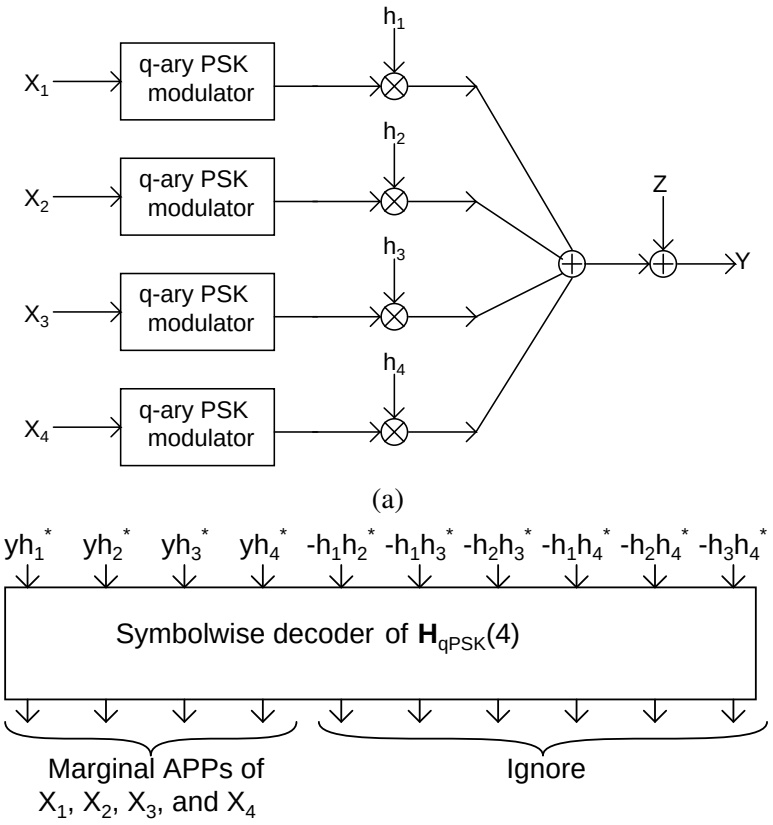

Figure 6.1: 1×4 MISO system. (a) The system model. (b) Demodulating the received symbol by using a symbolwise decoder.

of length N to $M \triangleq 2^N$ real amplitude values as in the naturally mapped PAM modulation, i.e.

$$\eta_N(\mathbf{x}) \triangleq \sum_{i=1}^{N} 2^{i-1}\beta(x_i), \tag{6.23}$$

where $\mathbf{x} = [x_1, x_2, \ldots, x_N]$ and $\beta(x)$ denotes binary antipodal mapping given in

$$\beta(x) \triangleq \begin{cases} 1, & x = 0 \\ -1, & x = 1 \end{cases}. \tag{6.24}$$

Assume that $\eta_N(\mathbf{X})$ is transmitted through a discrete additive Gaussian noise channel and Y is received. In other words,

$$Y = \eta_N(\mathbf{X}) + Z, \tag{6.25}$$

where $\mathbf{X} = [X_1, X_2, \ldots, X_N]$ and Z is a real Gaussian random variable with variance σ^2. Inserting the definition of $\eta_N(\mathbf{X})$ into (6.25) yields

$$Y = \sum_{i=1}^{N} 2^{i-1}\beta(X_i) + Z. \tag{6.26}$$

Since $\beta(X_i)$ is equal to $\mu_q(X_i)$ for $q = 2$ and 2 is a prime number, (6.26) is a special case of (6.2). Consequently, naturally mapped M-PAM detection is a special case of MISO detection of binary phase shift keying (BPSK) with channel coefficients $h_i = 2^{i-1}$. Hence, the parity check matrix of the code whose decoder can demodulate M-PAM is $\mathbf{H}_{2PSK}(\log_2 M)$. Since -1 is equal to 1 in the binary field, all of the minus ones in $\mathbf{H}_{2PSK}(\log_2 M)$ can be replaced with ones. The input vector that should applied to the decoder in order achieve demodulation of M-PAM is

$$\Big[\frac{y}{\sigma} \quad \frac{2y}{\sigma} \quad \cdots \quad \frac{2^{N-1}y}{\sigma} \quad -\frac{2^0 2^1}{\sigma} \quad -\frac{2^0 2^2}{\sigma} \quad -\frac{2^1 2^2}{\sigma} \quad -\frac{2^0 2^3}{\sigma} \quad -\frac{2^1 2^3}{\sigma} \quad -\frac{2^2 2^3}{\sigma} \quad \cdots \quad \cdots \quad -\frac{2^0 2^{N-1}}{\sigma} \quad -\frac{2^1 2^{N-1}}{\sigma} \quad \cdots -\frac{2^{N-2} 2^{N-1}}{\sigma}\Big],$$

where y denotes the received value.

Implementing an ML M-PAM detector by using the ML codeword decoder of the code with parity check matrix $\mathbf{H}_{2PSK}(\log_2 M)$ might not be practical since there are simpler ways to implement such a detector. However, implementing a soft output M-PAM detector by using the symbolwise decoder of the same code might be of practical importance.

Example 6.2 *This example shows how to compute marginal APPs of four bits which are modulated with naturally mapped 16-PAM and passed through an AWGN channel with PSD* $N_0/2 = \sigma^2$. *Constellation diagram of the naturally mapped 16-PAM is shown in Figure 6.2-a.*

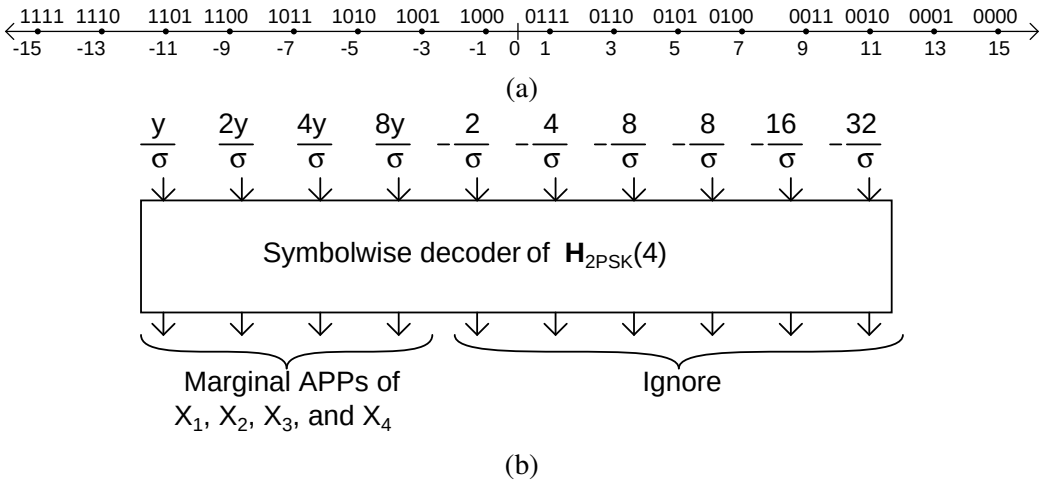

Figure 6.2: (a) Constellation diagram of naturally mapped 16-PAM modulation. (b) Computing marginal APPs from the received symbol by using the symbolwise decoder of $\mathbf{H}_{2PSK}(4)$.

The parity check matrix of the code whose symbolwise decoder can be used to compute marginal APPs of the individual bits is

$$\mathbf{H}_{2PSK}(4) = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{6.27}$$

Notice that this parity check matrix is a special case of the $\mathbf{H}_{qPSK}(4)$ *matrix given in the previous example for* $q = 2$*. Since* -1 *is equal to* 1 *in the binary field, minus ones in that matrix are replaced with plus ones.*

If the received value is denoted with y then the input vector that should be applied to this decoder is

$$\left[\frac{y}{\sigma} \quad \frac{2y}{\sigma} \quad \frac{4y}{\sigma} \quad \frac{8y}{\sigma} \quad -\frac{2}{\sigma} \quad -\frac{4}{\sigma} \quad -\frac{8}{\sigma} \quad -\frac{8}{\sigma} \quad -\frac{16}{\sigma} \quad -\frac{32}{\sigma} \right].$$

This example is illustrated in Figure 6.2.

6.4 Channel decoders as the detectors of gray mapped M-PAM

Naturally mapped M-PAM , whose detection by using a decoder is investigated in the previous section, suffers from the fact that more than one bits may differ between two adjacent symbols. This problem is overcome with the gray mapping in which a one bit differs between two adjacent symbols. In this section detection of gray mapped M-PAM by using a decoder is investigated. Let $\kappa_N(\mathbf{x})$ be a function from $\mathbb{F}_2^N$ to $\mathbb{R}$ which maps binary valued vectors of length N to $M \triangleq 2^N$ real amplitude values as in the gray mapped M-PAM, i.e.

$$\kappa_N(\mathbf{x}) \triangleq \sum_{i=1}^{N} 2^{i-1} \beta\left(\sum_{j=i}^{N} x_j\right), \tag{6.28}$$

where $\mathbf{x} = [x_1, x_2, \ldots, x_N]$ and the summation inside the $\beta(.)$ function takes places in $\mathbb{F}_2$. Unfortunately, due this summation inside the $\beta(.)$ function, detection of gray mapped M-PAM is not a special case MISO detection of BPSK as opposed to the detection of naturally mapped M-PAM. Hence, in order to determine the parity check matrix and inputs of the decoder to detect the M-PAM we need to obtain the canonical factorization of the joint a posteriori pmf.

Assume that $\kappa_N(\mathbf{X})$ is transmitted through a discrete additive Gaussian noise channel and Y is received. In other words,

$$Y = \kappa_N(\mathbf{X}) + Z, \tag{6.29}$$

where $\mathbf{X} = [X_1, X_2, \ldots, X_N]$ and Z is real Gaussian random variable with variance σ^2. Let $p(\mathbf{x})$ denote the joint a posteriori probability $\Pr\{\mathbf{X} = \mathbf{x}|Y = y\}$. The canonical factorization of $p(\mathbf{x})$ can be obtained by following the generic procedures explained in Chapter 3. However, the canonical factorization of $p(\mathbf{x})$ can be obtained more easily by exploiting the relation between $\kappa_N(\mathbf{x})$ and $\eta_N(\mathbf{x})$.

The relation between $\kappa_N(\mathbf{x})$ and $\eta_N(\mathbf{x})$ can be expressed as in

$$\kappa_N(\mathbf{x}) = \eta_N(\mathbf{x}\mathbf{G}(N)), \tag{6.30}$$

where $\mathbf{G}(N)$ is the $N \times N$ matrix defined as

$$\mathbf{G}(N) \triangleq \begin{bmatrix} 1 & 0 & \ldots & 0 \\ 1 & 1 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \ldots & 1 \end{bmatrix}. \tag{6.31}$$

Since $\mathbf{G}(N)$ is a reversible matrix, the canonical factorization of $p(\mathbf{x})$ can be derived from the canonical factorization of the APP $\Pr\{\mathbf{W} = \mathbf{w}|\eta_N(\mathbf{W}) + Z = y\}$ by following the discussion in Section 4.6.

Let $t(\mathbf{w})$ be the shorthand notation for the APP $\Pr\{\mathbf{W} = \mathbf{w}|\eta_N(\mathbf{W}) + Z = y\}$. Since $t(\mathbf{w})$ represents the APP in the naturally mapped M-PAM case, its canonical factorization is a special case of the canonical factorization given in (6.15) with channel coefficients $h_i = 2^{i-1}$ and $\mu_q(w) = \beta(w)$. The $\gamma(w;\rho,\sigma)$ function for the BPSK modulation is

$$\gamma(w;\rho,\sigma) = \exp\left(\frac{2\rho\beta(w)}{2\sigma^2}\right). \tag{6.32}$$

Consequently, $r_i(w)$ and $r_{i,j}(w)$ in this specific case of (6.15) are

$$r_i(w) = C_{\mathbb{F}_q}\left\{\gamma(w;2^{i-1}y,\sigma)\right\} = C_{\mathbb{F}_q}\left\{\exp\left(\frac{2^{i-1}y\beta(w)}{2\sigma^2}\right)\right\}, \tag{6.33}$$

$$r_{i,j}(w) = C_{\mathbb{F}_q}\left\{\gamma(w;-2^{i-1}2^{j-1},\sigma)\right\} = C_{\mathbb{F}_q}\left\{\exp\left(-\frac{2^{i+j-2}\beta(w)}{2\sigma^2}\right)\right\}. \tag{6.34}$$

Finally, the canonical factorization of $t(\mathbf{w})$ is

$$t(\mathbf{w}) = C_{\mathbb{F}_q^N}\left\{\prod_{i=1}^{N} r_i(\mathbf{f}_i\mathbf{w}^T)\prod_{j=2}^{N}\prod_{i=1}^{j-1} r_{i,j}(\mathbf{a}_{i,j}\mathbf{w}^T)\right\}. \tag{6.35}$$

Consequently, due to the discussion in Section 4.6 the canonical factorization of $p(\mathbf{x})$ is

$$p(\mathbf{x}) = C_{\mathbb{F}_q^N}\left\{\prod_{i=1}^{N} r_i(\mathbf{f}_i\mathbf{G}(N)^T\mathbf{x}^T)\prod_{j=2}^{N}\prod_{i=1}^{j-1} r_{i,j}(\mathbf{a}_{i,j}\mathbf{G}(N)^T\mathbf{x}^T)\right\}. \tag{6.36}$$

Let $\mathbf{b}_{i,j}$ defined as

$$\mathbf{b}_{i,j} \triangleq \sum_{k=i}^{j}\mathbf{f}_k. \tag{6.37}$$

$\mathbf{f}_i\mathbf{G}(N)^T$ and $\mathbf{a}_{i,j}\mathbf{G}(N)^T$ can be expressed by using $\mathbf{b}_{i,j}$ as

$$\mathbf{f}_i\mathbf{G}(N)^T = \mathbf{b}_{i,N}, \tag{6.38}$$

$$\mathbf{a}_{i,j}\mathbf{G}(N)^T = \mathbf{b}_{i,j-1}. \tag{6.39}$$

Consequently,

$$p(\mathbf{x}) = C_{\mathbb{F}_q^N}\left\{\prod_{i=1}^{N} r_i(\mathbf{b}_{i,N}\mathbf{x}^T)\prod_{j=2}^{N}\prod_{i=1}^{j-1} r_{i,j}(\mathbf{b}_{i,j-1}\mathbf{x}^T)\right\} \tag{6.40}$$

$$= C_{\mathbb{F}_q^N}\left\{r_N(\mathbf{f}_N\mathbf{x}^T)\prod_{i=1}^{N-1} r_{i,i+1}(\mathbf{f}_i\mathbf{x}^T)\prod_{i=1}^{N-1} r_i(\mathbf{b}_{i,N}\mathbf{x}^T)\prod_{j=2}^{N}\prod_{i=1}^{j-2} r_{i,j}(\mathbf{b}_{i,j-1}\mathbf{x}^T)\right\}. \tag{6.41}$$

This last form of the factorization clearly shows which parity check coefficient vectors are of weight two or more. Then, following the discussion in Section 5.6 the parity check matrix of this code whose decoder can be employed in the detection of gray mapped M-PAM is

$$\mathbf{H}_{GRAY}(N) \triangleq \begin{bmatrix} \mathbf{b}_{1,2} & \\ \mathbf{b}_{1,3} & \\ \mathbf{b}_{2,3} & \\ \vdots & \\ \mathbf{b}_{1,N} & \mathbf{I}_{\frac{N(N-1)}{2}\times\frac{N(N-1)}{2}} \\ \mathbf{b}_{2,N} & \\ \vdots & \\ \mathbf{b}_{N-1,N} & \end{bmatrix}. \tag{6.42}$$

Notice that the sizes of $\mathbf{H}_{GRAY}(N)$ and $\mathbf{H}_{qPSK}(N)$ are same.

The symbolwise and ML codeword decoders of the code with parity check matrix $\mathbf{H}_{GRAY}(N)$ can be designed for BPSK modulation and AWGN channel. In order to achieve the desired detection the inputs applied to this decoder should be a permuted version of the inputs applied for the naturally mapped detection since the canonical factorization of $p(\mathbf{x})$ is derived from the canonical factorization of $t(\mathbf{w})$. The first N of these inputs are

$$\left[-\frac{2^0 2^1}{\sigma} \quad -\frac{2^1 2^2}{\sigma} \quad \cdots \quad -\frac{2^{N-2}2^{N-1}}{\sigma} \quad \frac{2^{N-1}y}{\sigma}\right].$$

The last $N-1$ of these inputs are

$$\left[\frac{y}{\sigma} \quad \frac{2y}{\sigma} \quad \cdots \quad \frac{2^{N-2}y}{\sigma}\right].$$

The remaining $\frac{(N-2)(N-1)}{2}$) inputs in between are

$$\left[-\frac{2^0 2^2}{\sigma} \quad -\frac{2^0 2^3}{\sigma} \quad -\frac{2^1 2^3}{\sigma} \quad \cdots \quad -\frac{2^0 2^{N-1}}{\sigma} \quad -\frac{2^1 2^{N-1}}{\sigma} \quad \cdots \quad -\frac{2^{N-3}2^{N-1}}{\sigma}\right].$$

Example 6.3 *This example shows how to compute marginal APPs of four bits which are modulated with gray mapped 16-PAM and passed through an AWGN channel with PSD* $N_0/2 = \sigma^2$. *Constellation diagram of the 16-PAM modulation with gray mapping is shown in Figure 6.3-a. This example demonstrates an interesting property of demodulating gray mapped M-PAM modulation with decoders.*

The parity check matrix of the code whose symbolwise decoder can be used to compute marginal APPs of the individual bits is

$$\mathbf{H}_{GRAY}(4) = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{6.43}$$

If the received value is denoted with y then the input vector that should be applied to this decoder is

$$\begin{bmatrix} -\frac{2}{\sigma} & -\frac{8}{\sigma} & -\frac{32}{\sigma} & \frac{8y}{\sigma} & -\frac{4}{\sigma} & -\frac{8}{\sigma} & -\frac{16}{\sigma} & \frac{y}{\sigma} & \frac{2y}{\sigma} & \frac{4y}{\sigma} \end{bmatrix}.$$

It is well known that carrying out row operations on the parity check matrix of a code does not alter the code. Hence, we can carry out row operations on $\mathbf{H}_{GRAY}(4)$ *and obtain an alternative parity check matrix for the code. Let* $\mathbf{H}'$ *be the parity check matrix derived from* $\mathbf{H}_{GRAY}(4)$ *by adding the first row onto second and fourth rows and then adding sixth row onto fourth and fifth rows, i.e.*

$$\mathbf{H}' = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{6.44}$$

Since $\mathbf{H}'$ *and* $\mathbf{H}_{GRAY}(4)$ *are the parity check matrices of the same code, we can use the decoder designed for either* $\mathbf{H}'$ *or* $\mathbf{H}_{GRAY}(4)$ *to compute soft outputs in gray mapped 16-PAM modulation.*

Notice that all rows $\mathbf{H}'$ *are of weight* 3*. Moreover, four columns of* $\mathbf{H}'$ *are of weight* 3 *and the remaining six columns are of weight one.* $\mathbf{H}'$ *shares these properties with* $\mathbf{H}_{2PSK}(4)$*. Furthermore, let* $\mathbf{H}''$ *be the parity check matrix derived from* $\mathbf{H}'$ *by replacing the first column*

with fifth and fourth column with tenth, i.e.

$$\mathbf{H}'' = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{6.45}$$

$\mathbf{H}''$ *describes a code whose codewords are permuted form of the codewords of the code described by* $\mathbf{H}'$*. Hence, we can also use the decoder designed for* $\mathbf{H}''$ *to compute soft outputs in gray mapped 16-PAM modulation. In order to achieve this demodulation it is necessary to permute the inputs applied to the decoder designed for* $\mathbf{H}_{GRAY}(4)$ *before applying to the decoder designed for* $\mathbf{H}''$ *in the same order as the column permutations applied while passing from* $\mathbf{H}'$ *to* $\mathbf{H}''$*. Hence, the inputs that should be applied to this decoder are*

$$\begin{bmatrix} -\frac{4}{\sigma} & -\frac{8}{\sigma} & -\frac{32}{\sigma} & \frac{4y}{\sigma} & -\frac{2}{\sigma} & -\frac{8}{\sigma} & -\frac{16}{\sigma} & \frac{y}{\sigma} & \frac{2y}{\sigma} & \frac{8y}{\sigma} \end{bmatrix}.$$

The **interesting point** *in here is that* $\mathbf{H}''$ **is equal to** $\mathbf{H}_{2PSK}(4)$*. Therefore, the symbolwise decoder of the parity check matrix* $\mathbf{H}_{2PSK}(4)$ *can be used to compute marginal APPs for both naturally mapped and gray mapped 16-PAM modulation. The decoder can be configured to natural mapping or gray mapping by permuting the inputs. Computing the soft outputs in of gray mapped 16-PAM modulation depicted in Figure 6.3-c.*

The example above shows that the decoder of $\mathbf{H}_{2PSK}(4)$ can be used to demodulate both naturally mapped and gray mapped 16-PAM modulation. The following theorem states that this is true not only for 16-PAM but for any M-PAM modulation.

Theorem 6.1 *There exist a sequence of row operations such that performing these row operations on* $\mathbf{H}_{GRAY}(N)$ *leads to* $\mathbf{H}_{2PSK}(N)$ *with some columns permuted.*

A constructive proof is given in Appendix A.4.2.

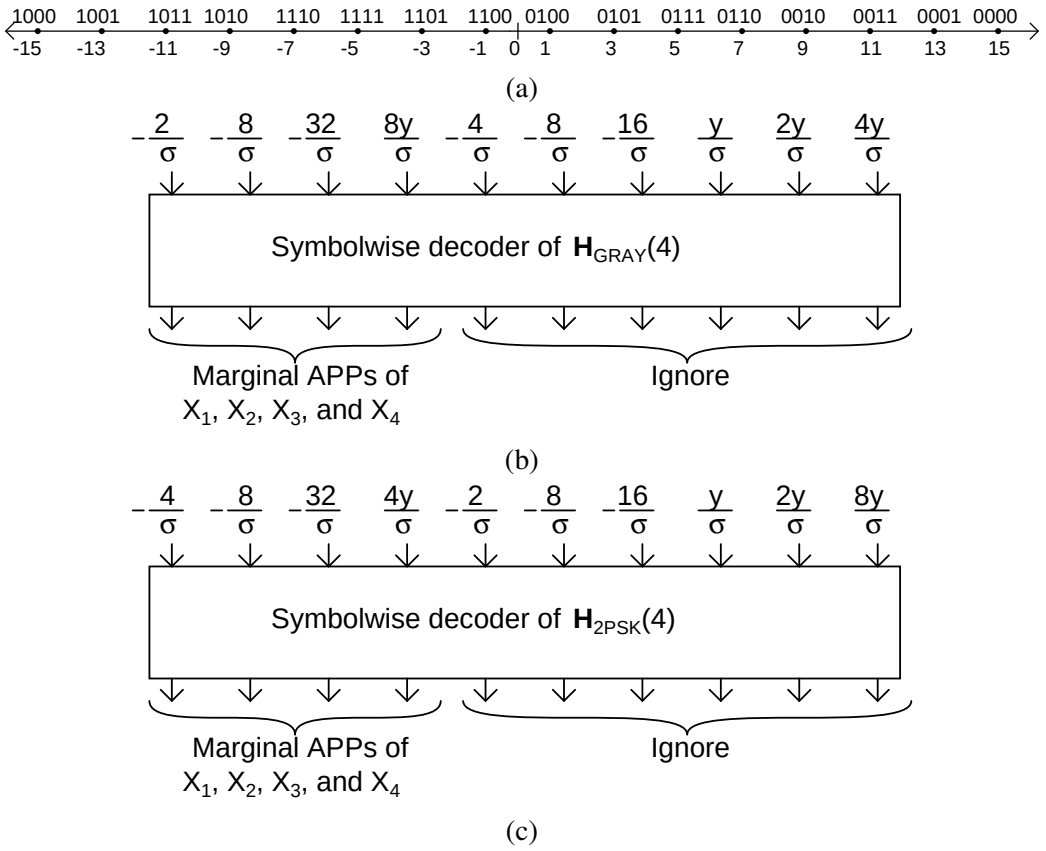

Figure 6.3: (a) Constellation diagram of gray mapped 16-PAM modulation. (b) Computing marginal APPs from the received symbol by using the symbolwise decoder of $\mathbf{H}_{GRAY}(4)$. (c) Computing marginal APPs by using the symbolwise decoder of $\mathbf{H}_{2PSK}(4)$.

6.5 MIMO detection by using channel decoders

In this section we show how to employ channel decoders for multiple-input multiple-output (MIMO) detection. The analysis is presented for QPSK modulation but is straightforward to extend method to any other PAM or QAM modulation.

6.5.1 System Model

Let a random vector $\mathbf{X}_k = [X_{2k-1}, X_{2k}]$ is mapped to a complex symbol W_k via the function $\nu(.)$ as in

$$W_k \triangleq \nu(\mathbf{X}_k), \tag{6.46}$$

where $\nu(.)$ represents the gray mapped QPSK modulation and defined as

$$\nu(\mathbf{x}) \triangleq \begin{cases} 1, & \mathbf{x} = [0 \quad 0] \\ j, & \mathbf{x} = [0 \quad 1] \\ -1, & \mathbf{x} = [1 \quad 1] \\ -j, & \mathbf{x} = [1 \quad 0] \end{cases}, \tag{6.47}$$

and j is the square root of -1. The constellation diagram of gray mapped QPSK modulation is shown in Figure 6.4. Furthermore, let a random vector $\mathbf{W} = [W_1, W_2, \ldots, W_{N_t}]^T$ is passed through an $N_r \times N_t$ MIMO channel with independent circularly symmetric Gaussian noise and the received vector is $\mathbf{Y}$. In other words,

$$\mathbf{Y} = \mathbf{H}_c\mathbf{W} + \mathbf{Z}, \tag{6.48}$$

where $\mathbf{H}_c$ is the $N_r \times N_t$ channel coefficient matrix, $\mathbf{Z}$ is the $N_r \times 1$ noise vector consisting of independent, zero mean, circularly symmetric normal distributed random variables of variance $2\sigma^2$.

ML MIMO detection is the task of determining the configuration $\mathbf{x}$ maximizes the likelihood function $\Pr\{\mathbf{Y} = \mathbf{y}|\mathbf{X} = \mathbf{x}\}$ where $\mathbf{X}$ is

$$\mathbf{X} \triangleq [\mathbf{X}_1 \quad \mathbf{X}_2 \quad \ldots \quad \mathbf{X}_{N_t}]. \tag{6.49}$$

We assume that all $\mathbf{X}$ is uniformly distributed. Hence, ML MIMO detection is equivalent to finding the configuration maximizing the APP $\Pr\{\mathbf{X} = \mathbf{x}|\mathbf{Y} = \mathbf{y}\}$. Soft output MIMO detection is the task of computing the marginal APPs $\Pr\{X_k = x|\mathbf{Y} = \mathbf{y}\}$.

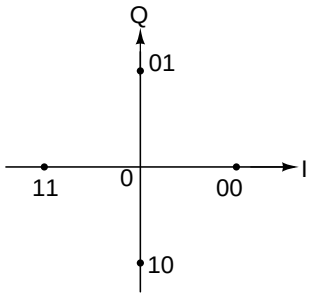

Figure 6.4: The QPSK constellation with gray mapping

6.5.2 The decoders which can be used in MIMO detection with QPSK signaling

The first step in determining the parity check matrix of the decoders which can be employed as MIMO demodulators is determining the canonical factorization of the APP. The APP $\Pr\{\mathbf{X} = \mathbf{x}|\mathbf{Y} = \mathbf{y}\}$ is

$$\Pr\{\mathbf{X} = \mathbf{x}|\mathbf{Y} = \mathbf{y}\} = C_{\mathbb{F}_q^{2N_t}} \left\{ \exp\left(-\frac{\|\mathbf{y} - \mathbf{H}_c\mathbf{w}\|^2}{2\sigma^2} \right) \right\} \tag{6.50}$$

where $\mathbf{x}$ is $[\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{N_t}]$, $\mathbf{x}_k$ is $[x_{2k-1}, x_{2k}]$, and $\mathbf{w}$ is $[\nu(\mathbf{x}_1), \nu(\mathbf{x}_2), \ldots, \nu(\mathbf{x}_{N_t})]^T$. As shown in Appendix A.4.3, this APP can factored as

$$\begin{aligned}
\Pr\{\mathbf{X} = \mathbf{x}|\mathbf{Y} = \mathbf{y}\} \propto & \prod_{k=1}^{N_t} \gamma\left(x_{2k-1}; \frac{\mathrm{Re}\{u_k\} + \mathrm{Im}\{u_k\}}{2}, \sigma \right) \gamma\left(x_{2k}; \frac{\mathrm{Re}\{u_k\} - \mathrm{Im}\{u_k\}}{2}, \sigma \right) \\
& \cdot \prod_{k=2}^{N_t} \prod_{l=1}^{k-1} \gamma\left(x_{2k-1} + x_{2l-1}; -\frac{\mathrm{Re}\{(\mathbf{R})_{k,l}\}}{2}, \sigma \right) \gamma\left(x_{2k-1} + x_{2l}; -\frac{\mathrm{Im}\{(\mathbf{R})_{k,l}\}}{2}, \sigma \right), \\
& \cdot \prod_{k=2}^{N_t} \prod_{l=1}^{k-1} \gamma\left(x_{2k} + x_{2l-1}; \frac{\mathrm{Im}\{(\mathbf{R})_{k,l}\}}{2}, \sigma \right) \gamma\left(x_{2k} + x_{2l}; -\frac{\mathrm{Re}\{(\mathbf{R})_{k,l}\}}{2}, \sigma \right)
\end{aligned} \tag{6.51}$$

where $\mathbf{R}$ and $\mathbf{u}$ are

$$\mathbf{R} \triangleq \mathbf{H}_c^H \mathbf{H}_c, \tag{6.52}$$

$$\mathbf{u} \triangleq \mathbf{H}_c^H \mathbf{y}, \tag{6.53}$$

and $(\mathbf{R})_{k,l}$ denotes k by l^{th} entry of $\mathbf{R}$ and u_k is the k^{th} component of $\mathbf{u}$.

The factorization above can be expressed by using the $\mathbf{a}_{k,l}$ vectors defined in (6.16) as

$$\begin{aligned}\Pr\{\mathbf{X}=\mathbf{x}|\mathbf{Y}=\mathbf{y}\} \propto & \prod_{k=1}^{N_t}\gamma\left(\mathbf{f}_{2k-1}\mathbf{x}^T;\frac{\mathrm{Re}\{u_k\}+\mathrm{Im}\{u_k\}}{2},\sigma\right)\gamma\left(\mathbf{f}_{2k}\mathbf{x}^T;\frac{\mathrm{Re}\{u_k\}-\mathrm{Im}\{u_k\}}{2},\sigma\right)\\ & \cdot\prod_{k=2}^{N_t}\prod_{l=1}^{k-1}\gamma\left(\mathbf{a}_{2k-1,2l-1}\mathbf{x}^T;-\frac{\mathrm{Re}\{(\mathbf{R})_{k,l}\}}{2},\sigma\right)\gamma\left(\mathbf{a}_{2k-1,2l}\mathbf{x}^T;-\frac{\mathrm{Im}\{(\mathbf{R})_{k,l}\}}{2},\sigma\right).\\ & \cdot\prod_{k=2}^{N_t}\prod_{l=1}^{k-1}\gamma\left(\mathbf{a}_{2k,2l-1}\mathbf{x}^T;\frac{\mathrm{Im}\{(\mathbf{R})_{k,l}\}}{2},\sigma\right)\gamma\left(\mathbf{a}_{2k,2l}\mathbf{x}^T;-\frac{\mathrm{Re}\{(\mathbf{R})_{k,l}\}}{2},\sigma\right)\end{aligned} \tag{6.54}$$

The only remaining step in the derivation of canonical factorization is to normalize all of the factor functions existing above. We omit this obvious step for the sake of neatness. This factorization leads to the parity check matrix of the decoder which can be used in MIMO detection given in

$$\mathbf{H}_{MIMO,QPSK}(N_t)\triangleq\begin{bmatrix}\mathbf{L}(1,N_t) & \\ \mathbf{L}(2,N_t) & \\ \cdots & \mathbf{I}_{2N_t(N_t-1)\times 2N_t(N_t-1)}\\ \mathbf{L}(N_t-1,N_t) & \end{bmatrix}, \tag{6.55}$$

where $\mathbf{L}(k,N_t)$ is

$$\mathbf{L}(k,N_t)\triangleq\begin{bmatrix}\mathbf{a}_{1,2k+1}\\ \mathbf{a}_{2,2k+1}\\ \cdots\\ \mathbf{a}_{2k,2k+1}\\ \mathbf{a}_{1,2k+2}\\ \mathbf{a}_{2,2k+2}\\ \cdots\\ \mathbf{a}_{2k,2k+2}\end{bmatrix}. \tag{6.56}$$

As in the previous sections we can use a decoder designed for BPSK modulation and AWGN channel for MIMO detection. The inputs that should be applied to this decoder to achieve MIMO detection are the first parameters after the semicolon divided by the second parameters of the $\gamma(.;.,.)$ functions in the factorization given in (6.54).

Example 6.4 *This example shows how to compute marginal APPs of four bits which are first modulated with QPSK modulation and passed through a* 2×2 *MIMO channel with channel*

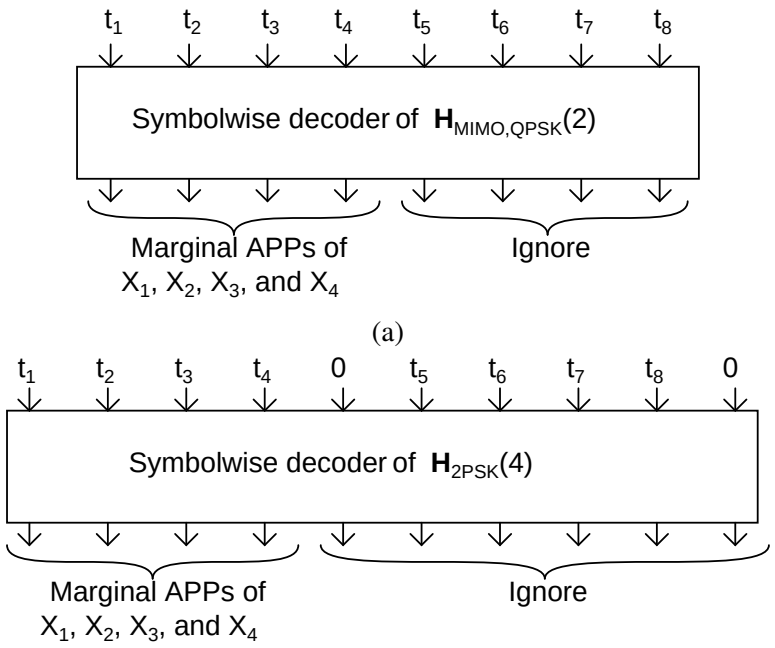

Figure 6.5: Computing the marginal APPs in a MIMO system by using two different decoders. (a) By using the decoder of $\mathbf{H}_{MIMO,QPSK}(2)$. (b) By using the decoder of $\mathbf{H}_{2PSK}(4)$.

coefficient matrix $\mathbf{H}_c$ *and noise variance* $2\sigma^2$. *The parity check matrix of the symbolwise decoder which can be used for this purpose is*

$$\mathbf{H}_{MIMO,QPSK}(2) = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{6.57}$$

Let the vector $\mathbf{t} = [t_1, t_2, \ldots, t_8]$ *be*

$$\mathbf{t} = \Big[\frac{\mathrm{Re}\{u_1\} + \mathrm{Im}\{u_1\}}{2\sigma} \quad \frac{\mathrm{Re}\{u_1\} - \mathrm{Im}\{u_1\}}{2\sigma} \quad \frac{\mathrm{Re}\{u_2\} + \mathrm{Im}\{u_2\}}{2\sigma} \quad \frac{\mathrm{Re}\{u_2\} - \mathrm{Im}\{u_2\}}{2\sigma} \\ - \frac{\mathrm{Re}\{(\mathbf{R})_{1,2}\}}{2\sigma} \quad - \frac{\mathrm{Im}\{(\mathbf{R})_{1,2}\}}{2\sigma} \quad \frac{\mathrm{Im}\{(\mathbf{R})_{1,2}\}}{2\sigma} \quad - \frac{\mathrm{Re}\{(\mathbf{R})_{1,2}\}}{2\sigma} \Big], \tag{6.58}$$

where $\mathbf{R} = \mathbf{H}_c^H \mathbf{H}_c$ *and* $\mathbf{u} = \mathbf{H}_c \mathbf{y}$. *This* $\mathbf{t}$ *vector is the vector that must be applied to the decoder.*

Notice that $\mathbf{H}_{MIMO,QPSK}(2)$ *is a sub-matrix of* $\mathbf{H}_{2PSK}(4)$. *Therefore, the symbolwise decoder of* $\mathbf{H}_{2PSK}(4)$ *can also be used to compute marginal APP probabilities in MIMO detection. The*

inputs that must be applied in this case are given below.

$$\left[\frac{\mathrm{Re}\{u_1\}+\mathrm{Im}\{u_1\}}{2\sigma} \quad \frac{\mathrm{Re}\{u_1\}-\mathrm{Im}\{u_1\}}{2\sigma} \quad \frac{\mathrm{Re}\{u_2\}+\mathrm{Im}\{u_2\}}{2\sigma} \quad \frac{\mathrm{Re}\{u_2\}-\mathrm{Im}\{u_2\}}{2\sigma}\right.$$
$$\left. 0 \quad -\frac{\mathrm{Re}\{(\mathbf{R})_{1,2}\}}{2\sigma} \quad -\frac{\mathrm{Im}\{(\mathbf{R})_{1,2}\}}{2\sigma} \quad \frac{\mathrm{Im}\{(\mathbf{R})_{1,2}\}}{2\sigma} \quad -\frac{\mathrm{Re}\{(\mathbf{R})_{1,2}\}}{2\sigma} \quad 0\right], \tag{6.59}$$

Notice that we added two zeros to the input vector when compared to the vector $\mathbf{t}$*. These zeros correspond the missing columns in* $\mathbf{H}_{MIMO,QPSK}(2)$ *when compared to* $\mathbf{H}_{2PSK}(4)$*. Computing the marginal APPs with these two decoders is depicted in Figure 6.5.*

It is worth emphasizing that in Examples 6.2, 6.3, and 6.4 the decoder of $\mathbf{H}_{2PSK}(4)$ is used for *three different purposes*.

6.6 Usage of decoders of tail biting convolutional codes as approximate MIMO detectors

Trellis representation is mainly used for representing convolutional codes. However, it is also possible to represent block codes with trellises [33]. Block codes can also be represented with a special type of trellis which is the tail biting trellis. Maximum trellis width in a tail biting trellis might be as low as the square root of the maximum width of the ordinary trellis representing the same code [11, 6].

If a block code has a parity check matrix as in the form given below

$$\mathbf{H} = \begin{bmatrix} ((\mathbf{L}_{r\times c}))_0 & \\ ((\mathbf{L}_{r\times c}))_1 & \mathbf{I}_{rc\times rc} \\ \cdots & \\ ((\mathbf{L}_{r\times c}))_{c-1} & \end{bmatrix}, \tag{6.60}$$

where $\mathbf{L}_{r\times c}$ is any $r\times c$ matrix and $((\mathbf{L}))_i$ denotes cyclically shifting the columns of $\mathbf{L}$ towards right i times, then it is called a tail biting convolutional code of rate $1/(r+1)$. For instance, the Golay code is of this type [11]. Tail biting convolutional codes can be encoded by the encoders of the convolutional codes by applying the data bits cyclically.

The tail biting convolutional codes have simple approximate decoders enjoying low complexity [11, 6]. Hence, there are many studies and standards, such as LTE, exploiting this reduction in complexity and simplicity of the tail biting trellises. Even an analog implementation of such a decoder is proposed in [14].

In this section we show that the MIMO detection problem can be handled by the decoder of a tail biting convolutional code. The characteristics of this code depend on the number of transmitting antennae and modulation used. We are going to analyze the MIMO detectors for QPSK modulation as we did in the previous section, although it is possible to generalize the technique to other QAM and PAM modulations as well.

We are going to use the same channel model and notation as in the previous section. That model lead us the parity check matrix $\mathbf{H}_{MIMO,QPSK}(N)$ given in (6.55). This parity check matrix hardly looks like the parity check matrix of a tail biting convolutional code.

Let a permutation matrix $\mathbf{P}$ is defined as in

$$\mathbf{P} \triangleq \begin{bmatrix} \mathbf{f}_1^T & \mathbf{f}_3^T & \dots & \mathbf{f}_{2N_t-1}^T & \mathbf{f}_2^T & \mathbf{f}_4^T & \dots & \mathbf{f}_{2N}^T \end{bmatrix}. \tag{6.61}$$

Furthermore, let $\mathbf{V}$ be obtained by permuting $\mathbf{X}$ as in

$$\mathbf{V} \triangleq \mathbf{X}\mathbf{P}. \tag{6.62}$$

Since $\mathbf{V}$ is a permutation of $\mathbf{X}$, maximizing (marginalizing) the APP $\Pr\{\mathbf{X} = \mathbf{x}|\mathbf{Y} = \mathbf{y}\}$ is equivalent to maximizing (marginalizing) the APP $\Pr\{\mathbf{V} = \mathbf{v}|\mathbf{Y} = \mathbf{y}\}$. Let $t(\mathbf{v})$ be a shorthand notation for $\Pr\{\mathbf{V} = \mathbf{v}|\mathbf{Y} = \mathbf{y}\}$. Then the factorization of $t(\mathbf{v})$ can be derived from the factorization (6.54) as

$$\begin{aligned} t(\mathbf{v}) \propto & \prod_{k=1}^{N_t} \gamma\left(\mathbf{f}_{2k-1}\mathbf{P}\mathbf{v}^T; \frac{\mathrm{Re}\{u_k\} + \mathrm{Im}\{u_k\}}{2}, \sigma\right) \gamma\left(\mathbf{f}_{2k}\mathbf{P}\mathbf{v}^T; \frac{\mathrm{Re}\{u_k\} - \mathrm{Im}\{u_k\}}{2}, \sigma\right) \\ & \cdot \prod_{k=2}^{N_t} \prod_{l=1}^{k-1} \gamma\left(\mathbf{a}_{2k-1,2l-1}\mathbf{P}\mathbf{v}^T; -\frac{\mathrm{Re}\{(\mathbf{R})_{k,l}\}}{2}, \sigma\right) \gamma\left(\mathbf{a}_{2k-1,2l}\mathbf{P}\mathbf{v}^T; -\frac{\mathrm{Im}\{(\mathbf{R})_{k,l}\}}{2}, \sigma\right), \\ & \cdot \prod_{k=2}^{N_t} \prod_{l=1}^{k-1} \gamma\left(\mathbf{a}_{2k,2l-1}\mathbf{P}\mathbf{v}^T; \frac{\mathrm{Im}\{(\mathbf{R})_{k,l}\}}{2}, \sigma\right) \gamma\left(\mathbf{a}_{2k,2l}\mathbf{P}\mathbf{v}^T; -\frac{\mathrm{Re}\{(\mathbf{R})_{k,l}\}}{2}, \sigma\right) \end{aligned} \tag{6.63}$$

since $(\mathbf{P}^{-1})^T = \mathbf{P}$. Consequently, a parity check matrix whose ML codeword (symbolwise) decoder can be employed in maximization (marginalization) of $\Pr\{\mathbf{V} = \mathbf{v}|\mathbf{Y} = \mathbf{y}\}$ is

$$\mathbf{H}_{\mathbf{V}}(N_t) \triangleq \begin{bmatrix} \mathbf{B}(N_t) & \mathbf{I}_{2N_t(N_t-1)\times 2N_t(N_t-1)} \end{bmatrix} \tag{6.64}$$

where $\mathbf{B}(N)$ is

$$\mathbf{B}(N) \triangleq \begin{bmatrix} \mathbf{L}(1,N)\mathbf{P} \\ \mathbf{L}(2,N)\mathbf{P} \\ \dots \\ \mathbf{L}(N-1,N)\mathbf{P} \end{bmatrix}. \tag{6.65}$$

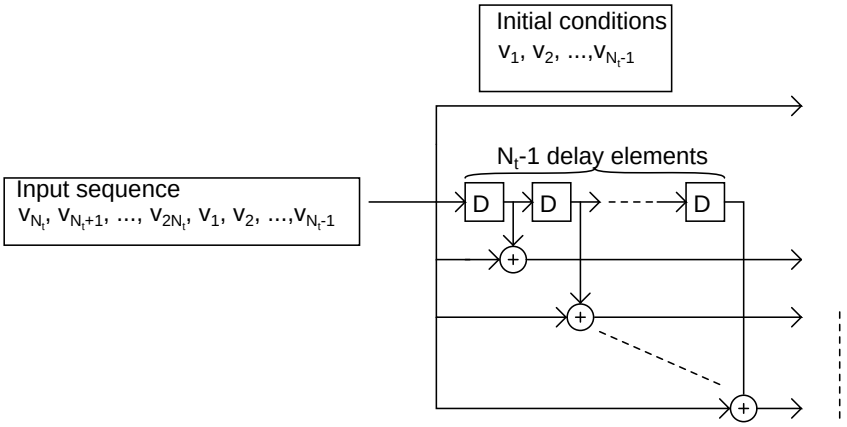

Figure 6.6: The encoder of the tail biting convolutional code whose decoder can be used as the detector MIMO system with N_t transmit and N_r receiving antennae. Notice that as opposed to ordinary convolutional encoders the encoder does not initiate from the all zero state. Tail biting nature of the decoder arises from the fact that after all the input sequence is applied the decoder returns to its initial condition.

Other alternative parity check matrices whose decoder can be employed in performing inference on $\Pr\{\mathbf{V} = \mathbf{v}|\mathbf{Y} = \mathbf{y}\}$ are in the form of

$$\left[\mathbf{B}'(N_t) \quad \mathbf{I}_{2N_t(N_t-1)\times 2N_t(N_t-1)}\right],$$

where $\mathbf{B}'(N_t)$ is derived from $\mathbf{B}(N_t)$ by permuting rows (not columns this time). Fortunately, there exists a special row permutation which forms $\mathbf{B}(N_t)$ into the form given in

$$\mathbf{B}_{TB}(N_t) \triangleq \begin{bmatrix} ((\mathbf{L}_{TB}(N_t) & \mathbf{0}_{(N_t-1)\times N_t}))_0 \\ ((\mathbf{L}_{TB}(N_t) & \mathbf{0}_{(N_t-1)\times N_t}))_1 \\ & \cdots \\ ((\mathbf{L}_{TB}(N_t) & \mathbf{0}_{(N_t-1)\times N_t}))_{2N_t} \end{bmatrix}, \tag{6.66}$$

where $\mathbf{L}_{TB}(N)$ is

$$\mathbf{L}_{TB}(N) \triangleq \left[\mathbf{I}_{(N_t-1)\times(N_t-1)} \quad \mathbf{1}_{(N_t-1)\times 1}\right]. \tag{6.67}$$

Consequently, the decoders of the parity check matrix given in

$$\mathbf{H}_{TB,MIMO}(N_t) = \left[\mathbf{B}_{TB}(N_t) \quad \mathbf{I}_{2N_t(N_t-1)\times 2N_t(N_t-1)}\right] \tag{6.68}$$

can be employed in performing inference on $\Pr\{\mathbf{V} = \mathbf{v}|\mathbf{Y} = \mathbf{y}\}$. $\mathbf{H}_{TB,MIMO}(N_t)$ is the parity check matrix of the tail biting convolutional code of rate $(1/(N_t))$ and constraint length N_t, whose encoder is shown in Figure 6.6.

Example 6.5 *In this example we are demonstrate that* $\mathbf{B}_{TB}(N_t)$ *can be derived from* $\mathbf{B}(N_t)$ *by permuting rows for cases* $N_t = 2$ *and* $N_t = 3$.

For $N_t = 2$, $\mathbf{B}(N_t)$ *is equal to* $\mathbf{L}(1,2)\mathbf{P}$. *By (6.56),* $\mathbf{L}(1,2)$ *is*

$$\mathbf{L}(1,2) = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix}. \tag{6.69}$$

Consequently, $\mathbf{B}(2)$ *is*

$$\mathbf{B}(2) = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}. \tag{6.70}$$

Changing the places of third and fourth rows gives $\mathbf{B}_{TB}(2)$, *which is*

$$\mathbf{B}_{TB}(2) = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \end{bmatrix}. \tag{6.71}$$

The Tanner graph of the resulting $\mathbf{H}_{TB,MIMO}(2) = [\mathbf{B}_{TB}(2) \quad \mathbf{I}_{4\times4}]$ *is shown in Figure 6.7-a.*

For $N_t = 2$, $\mathbf{B}(N_t)$ *is equal to* $\begin{bmatrix} \mathbf{L}(1,3) \\ \mathbf{L}(2,3) \end{bmatrix}\mathbf{P}$ *where* $\begin{bmatrix} \mathbf{L}(1,3) \\ \mathbf{L}(2,3) \end{bmatrix}$ *is*

$$\begin{bmatrix} \mathbf{L}(1,3) \\ \mathbf{L}(2,3) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix}. \tag{6.72}$$

Then $\mathbf{B}(3)$ *is*

$$\mathbf{B}(3) = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}. \tag{6.73}$$

Finally carrying the 5^{th}, 7^{th}, 2^{nd}, 6^{th}, 8^{th}, 4^{th}, 10^{th}, 12^{th}, 3^{rd}, 9^{th}, 11^{th}, *and* 1^{st} *rows to* 1^{st}, 2^{nd}, $\ldots$, 12^{th} *rows gives* $\mathbf{B}_{TB}(3)$ *as in*

$$\mathbf{B}_{TB}(3) = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{6.74}$$

The Wiberg style Tanner graph of the resulting $\mathbf{H}_{TB,MIMO}(3) = [\mathbf{B}_{TB}(3) \quad \mathbf{I}_{12\times 12}]$ *is shown in Figure 6.7-b.*

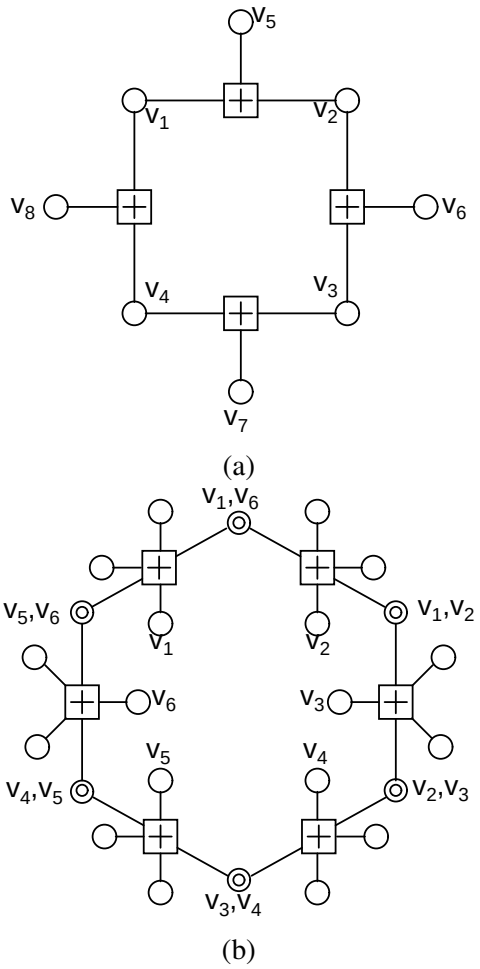

Figure 6.7: The Tanner graphs of $\mathbf{H}_{TB,MIMO}(N_t)$ for $N_t = 2$ and $N_t = 3$. (a) Tanner graph of $\mathbf{H}_{TB,MIMO}(2)$. (b) Wiberg style Tanner graph of $\mathbf{H}_{TB,MIMO}(3)$

6.6.1 Using the decoding algorithms of tail biting convolutional codes for MIMO detection

Since a tail biting trellis does not have a starting or ending state, Viterbi and BCJR algorithms cannot be run on such trellises directly. To process a tail biting trellis with Viterbi algorithm we need to run the Viterbi algorithm ν times on the trellis where ν denotes the trellis width. In each run, the Viterbi algorithm determines a candidate path which is the most probable path among the paths starting and ending on a certain state on the trellis. Then the most probable path can be chosen among the ν candidate paths. Since the complexity of each running of the Viterbi algorithm is $O(L\nu)$, where L denotes the length of the trellis, the complexity of determining the most possible path with Viterbi algorithm is $O(L\nu^2)$. Recall that the complexity would be $O(L\nu)$ if the trellis were an ordinary trellis. Similar arguments are true for the BCJR algorithm as well.

The complexity of ML codeword and exact symbolwise decoders of $\mathbf{H}_{TB,MIMO}(N_t)$ is $O(N_t 2^{2N_t})$ as explained in the previous paragraph. The complexity of the trivial MIMO detection algorithm is $O(2^{2N_t})$. Hence, using the exact decoders of $\mathbf{H}_{TB,MIMO}(N_t)$ for MIMO detection does not make sense.

Fortunately, tail biting convolutional codes have an *approximate* symbolwise decoder. This decoder operates by running BCJR algorithm on the tail biting trellis *iteratively*. Equivalently, this decoder can be viewed as the iterative sum-product algorithm running on the Wiberg style Tanner graph an example of which is shown in Figure 6.7-b. Usually, a few iterations are sufficient to converge [6]. We propose implementing an approximate soft output MIMO detector by using this approximate symbolwise as the decoder $\mathbf{H}_{TB,MIMO}(N_t)$. Such a MIMO detector is also capable of using any a priori information available since it uses the BCJR algorithm. The block diagram of this *approximate* soft output MIMO detector is shown in Figure 6.8.

6.6.2 Complexity issues

There are two subtasks when the decoder mentioned above is employed as an approximate soft ouput MIMO detector. These tasks are the computation of the inputs applied to the decoder and processing the decoder trellis.

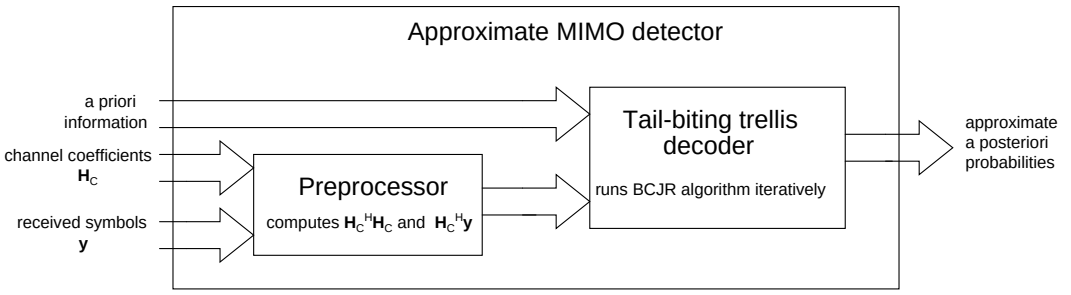

Figure 6.8: Block diagram of the proposed approximate soft output MIMO detector which uses the approximate decoder of a tail biting convolutional code.

As explained in Section 6.5, the inputs that must be applied to the decoder of $\mathbf{H}_{TB,MIMO}$ are the components of $\mathbf{u}$ and the entries of $\mathbf{R}$ defined in (6.52) and (6.53) respectively. The computation of $\mathbf{u}$ has a complexity $O(N_r N_t)$ whereas the computation of $\mathbf{R}$ has a complexity $O(N_t^2 N_r)$.

Processing the decoding trellis with the BCJR algorithm has a complexity $O(N_t 2^{N_t})$. This complexity is almost the square root of the complexity of the trivial ML and soft output MIMO detectors which is $O(2^{2N_t})$. From a computer scientific point of view, this last component of the complexity might be dominant to the complexity of the computation of $\mathbf{R}$. However, in an engineering point of view computing $\mathbf{R}$ is a more computationally demanding task than processing the decoding trellis for two reasons. First, in a practical scenario N_r and N_t is eight at most. Hence, $N_t 2^{N_t}$ and $N_t^2 N_r$ are comparable in practical scenarios. Second, the decoding trellis processing involves only additions and maximizations[3] whereas computing $\mathbf{R}$ involves complex multiplications which require much more complex hardware than addition. Therefore, computing $\mathbf{R}$ is the most computationally demanding subtask of the proposed method. However, it should be noted that $\mathbf{R}$ is computed only once for a constant $\mathbf{H}_c$.

The proposed technique, which employs a tail biting decoder as the MIMO detector, is comparable to other sub optimal methods such as minimum mean square error (MMSE) or zero forcing (ZF) detectors in terms of hardware complexity which both have a complexity $O(N^3)$ if $N_t = N_r = N$. Furthermore, other sub optimal methods require matrix inversion. Although, matrix inversion have complexity $O(N^3)$, it requires complex number divisions which require even more complex hardware than multiplication. Hence, the proposed technique still has an advantage in terms of hardware complexity over MMSE and ZF detectors.

[3] We assume Max-Log-MAP approximation is used for the BCJR algorithm running on the trellis.

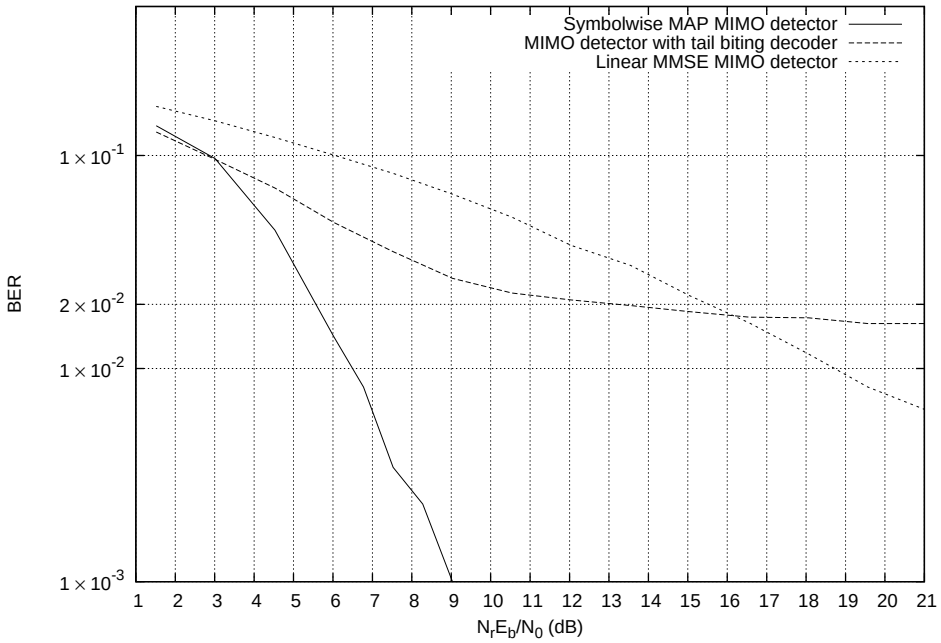

Figure 6.9: BER performances of the MIMO detector using the decoder of a tail biting convolutional code, the symbolwise MAP MIMO detector, and the linear MMSE MIMO detector in a Rayleigh fading 8×8 MIMO channel.

6.6.3 Simulation Results

We simulated the proposed approximate soft output MIMO detector for the 8×8 Rayleigh fading MIMO channel. In this channel entries of $\mathbf{H}_c$ are independent, zero-mean, circularly symmetric Gaussian random variables where the variances of the real and imaginary parts are $1/2$. We assumed that $\mathbf{H}_c$ changes for every transmitted MIMO symbol and perfectly known at the receiver side. The noise vector added at the receiver also consists of independent, zero-mean, circularly symmetric Gaussian random variables where the variances of the real and imaginary parts are $N_0/2$. The signal to noise ratio (SNR) per receiving antenna is E_b/N_0. Since there are N_r receiving antennae in a MIMO system the convention is to use $N_r E_b/N_0$ as SNR [37].

The bit error rate (BER) performance of the proposed algorithm is shown in Figure 6.9. These results show that the proposed method has an unexpected poor performance when compared to the symbolwise MAP MIMO detector. Moreover, the proposed method exhibits an error floor as early as 2×10^{-2} level. The performance of the proposed algorithm is better than the

linear minimum mean square error (MMSE) MIMO detector [43] until 16 dB. After 16 dB the performance of the linear MMSE becomes better due to the early error floor of the proposed MIMO detector. We provide some comments on this unexpected performance in the next section and propose an improvement in Section 6.6.5.

6.6.4 Comments on the convergence of the sum-product algorithm on factor graphs with a single cycle

The Wiberg style Tanner graph that represents the tail biting trellis contains only a single loop, as in Figure 6.7-a. There are many studies in the sum-product algorithm literature which claim that the sum-product algorithm running on Tanner graph with a single cycle always converges such as [38, 39, 40]. These studies also claim that the approximate marginals computed by the sum-product algorithm is close to the exact marginals when the sum-product runs on these graphs. According to these studies, our proposed MIMO detector was supposed to converge at all times and it was expected to yield good results. However, our empirical results shown in Figure 6.9 do not agree with these expectations.

Our experimental results verify that the sum-product algorithm running on a Tanner factor graph with a single cycle always converges. However, in some cases this convergence require as few as two or three iterations to converge whereas in some other rare cases it might require thousands of iterations. A detailed analysis of the experimental results shows that the relatively high error floor in Figure 6.9 is caused by the cases in which the sum-product algorithm requires thousands of iterations to converge. Therefore, the sum-product algorithm produces good approximations of the exact marginals only if it converges in a few iterations. Otherwise, the results generated by the sum-product algorithm is not a good approximation. We provide a numerical example in which sum-product algorithm requires thousands of iterations to converge below.

Example 6.6 *We provide the example for the Tanner graph shown in Figure 6.7-a which is a factor graph with just a single cycle and contains only binary variable nodes. Let the inputs applied to the decoder represented by the Tanner graph shown in Figure 6.7-a designed for BPSK modulation and AWGN channel be*

$$[-55 \quad 60 \quad -25 \quad -20 \quad 40 \quad 55 \quad 40 \quad -55] \tag{6.75}$$

If one runs the sum product algorithm on the Tanner graph shown in Figure 6.7-a with these inputs, it can be observed that the sum-product algorithm achieves a reasonable convergence at least after 3000 *iterations. Such an input settings can be observed in a scenario in which that decoder is employed as a MIMO detector for a* 2×2 *channel with coefficients*

$$\mathbf{H}_c = \begin{bmatrix} 1.5j & 1 - 0.5j \\ 1 + 0.5j & -0.5 - 1.5j \end{bmatrix}$$

and a sequence $[-j, 1]$ *is transmitted when noise has a variance* $\sigma^2 = 0.01$.

We would like to note that the likelihoods given above are very unlikely to be observed in a real channel decoding problem. Therefore, such likelihoods is probably never observed in [38, 39, 40]. Hence, they claimed that the sum-product algorithm produces good approximations for exact marginals if the sum-product algorithm converges. Unfortunately, this claim is not quite true as this counter example shows.

Even if the sum-product algorithm produced good approximations in cases requiring thousands of iterations to converge, a practical MIMO detection algorithm cannot wait that much to complete the demodulation of a single MIMO symbol. Therefore, this late convergence problem requires a solution to develop a practical MIMO detection algorithm with tail biting decoders which we provide in the next section.

6.6.5 Performance Improvements by using tail biting convolutional codes of longer constraint length

Recall that the tail biting decoder of $\mathbf{H}_{TB,MIMO}(N_t)$ is used for performing inference on $\Pr\{\mathbf{V} = \mathbf{v}|\mathbf{Y} = \mathbf{y}\}$ where $\mathbf{V}$ was a permutation of $\mathbf{X}$ given by (6.62). This decoder can only perform inference for this specific permutation of $\mathbf{X}$.

We define an *extended* version of parity check matrix $\mathbf{H}_{TB,MIMO}(N_t)$ as in

$$\mathbf{H}_{ETB,MIMO}(N_t) \triangleq \left[\mathbf{C}_{TB}(N_t) \quad \mathbf{I}_{2N_t^2 \times 2N_t^2}\right], \tag{6.76}$$

where $\mathbf{C}_{TB}(N_t)$ is

$$\mathbf{C}_{TB}(N_t) \triangleq \begin{bmatrix} ((\mathbf{L}_{TB}(N_{t+1}) & \mathbf{0}_{N_t\times(N_t-1)}))_0 \\ ((\mathbf{L}_{TB}(N_{t+1}) & \mathbf{0}_{N_t\times(N_t-1)}))_1 \\ & \cdots \\ ((\mathbf{L}_{TB}(N_{t+1}) & \mathbf{0}_{N_t\times(N_t-1)}))_{2N_t} \end{bmatrix}. \tag{6.77}$$

Notice that $\mathbf{H}_{ETB,MIMO}(N_t)$ is the parity check matrix of a tail biting convolutional code of rate $1/(N_t + 1)$ and of constraint length $N_t + 1$ and can be derived from $\mathbf{H}_{TB,MIMO}(N_t)$ by adding $2N_t$ more parity checks.

As opposed to the decoder of $\mathbf{H}_{TB,MIMO}(N_t)$, which can perform inference only on $\Pr\{\mathbf{V} = \mathbf{v}|\mathbf{Y} = \mathbf{y}\}$, the decoder of $\mathbf{H}_{ETB,MIMO}(N_t)$ can be used to perform inference on $\Pr\{\mathbf{V}_A = \mathbf{v}|\mathbf{Y} = \mathbf{y}\}$ where $\mathbf{V}_A$ is *any permutation* of $\mathbf{X}$.

An improved soft output MIMO detector can be implemented by using the approximate symbolwise detector of $\mathbf{H}_{ETB,MIMO}(N_t)$ instead of $\mathbf{H}_{TB,MIMO}(N_t)$. The main advantage of the detector with extended tail biting decoder when compared original tail biting decoder is that it can work with any permutation of $\mathbf{X}$. Moreover, a certain permutation can work better for a given $\mathbf{H}_c$ and noise realization while another permutation can work better with another $\mathbf{H}_c$ and noise realization. This flexibility comes at the cost of increasing trellis processing complexity by two which is acceptable.

We propose a soft output MIMO detection algorithm by using the approximate symbolwise decoder of the extended tail biting code as follows.

1. Select a permutation $\mathbf{P}_A$ from a set $\mathcal{P}$ of permutations.
2. Apply the inputs properly permuted with the permutation $\mathbf{P}_A$ to the approximate symbolwise decoder of $\mathbf{H}_{ETB,MIMO}(N_t)$.
3. Run the BCJR algorithm iteratively on the tail biting trellis until it converges or a maximum number of iterations reached.
4. If the iterative BCJR algorithm converges declare its result as the output of the MIMO detector and halt.
5. If the iterative BCJR algorithm does not converge select another permutation $\mathbf{P}_A$ from $\mathcal{P}$ and goto Step 2. If there is not any remaining permutation in $\mathcal{P}$ then declare a failure.

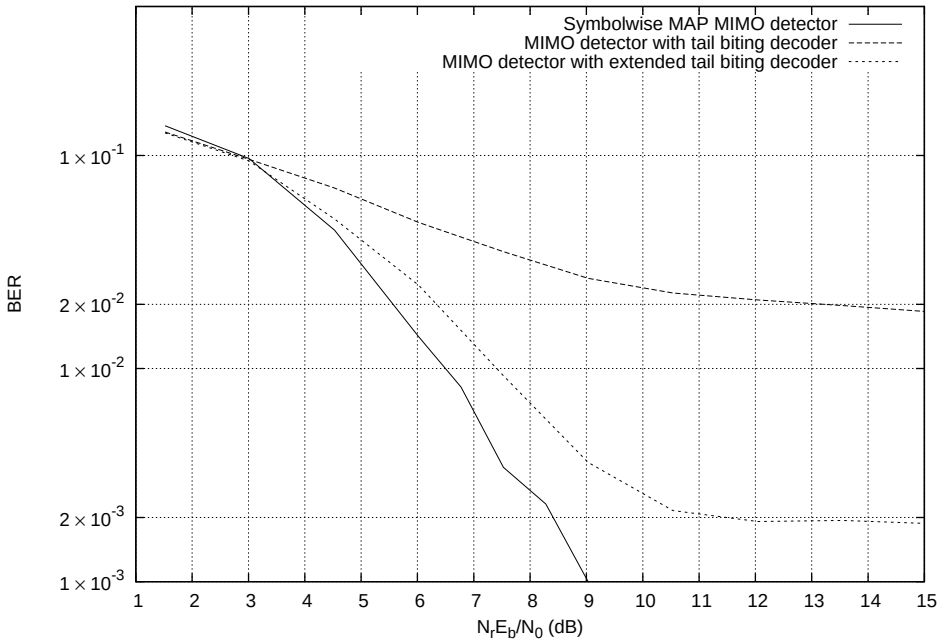

Figure 6.10: BER performance of the MIMO detector using the extended tail biting decoder with different permutations together with the MIMO detector with the normal tail biting decoder and symbolwise MAP MIMO detector in Rayleigh fading 8×8 channel.

.

We tested this algorithm on the 8×8 MIMO channel described in Section 6.6.3. The set $\mathcal{P}$ we used in this simulations consists of 15 specific permutations among 16! possible permutations. These permutations are given Appendix A.4.4. BER performance of this MIMO detector is given in Figure 6.10. These results show that the MIMO detector using the extended tail biting decoder improves the error floor performance by an order of magnitude. Furthermore, the BER performance before reaching the error floor is also improved significantly. The improved MIMO detector is just 2dB away from the optimum algorithm when it reaches the error floor.

Recall that this MIMO detector is capable of using a priori information and produces soft output. Hence, it can be easily used in a iterative detection-decoding scheme. In order estimate the possible performance of the improved MIMO detector in such an iterative scheme, we computed extrinsic information transfer (EXIT) curves [35, 36, 37]. The area under the EXIT curve of a MIMO detector is an approximate estimation of the maximum possible rate of the code which can be used in an iterative detection-decoding scheme and can achieve arbitrarily small error rate. In this aspect the area under exact soft output MIMO detector is an

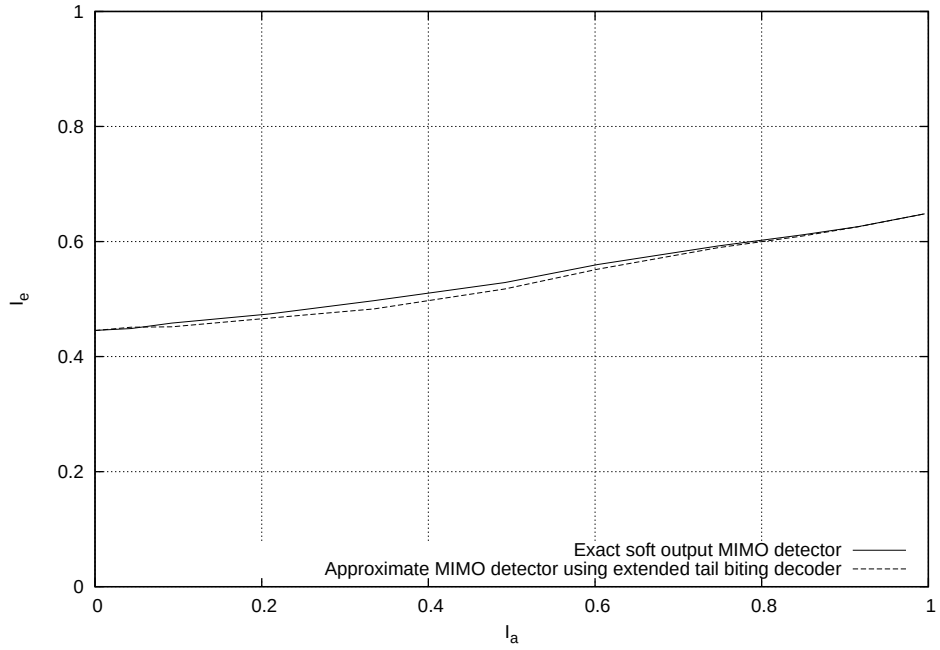

Figure 6.11: EXIT curves of the approximate MIMO detector using extended tail biting decoder and the exact soft output MIMO detector at $N_r E_b/N_0 = -0.96$dB

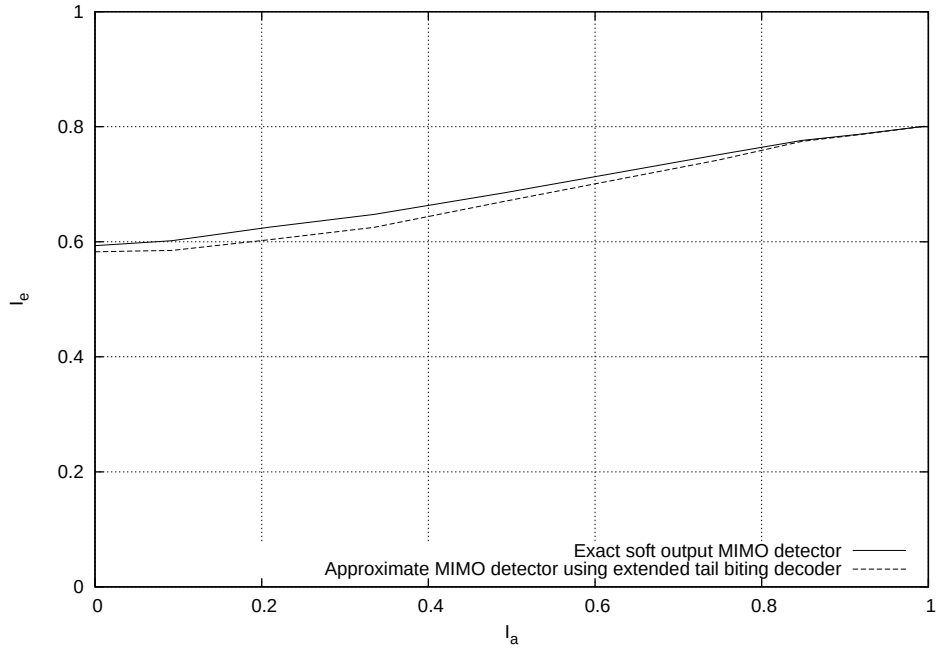

Figure 6.12: EXIT curves of the approximate MIMO detector using extended tail biting decoder and the exact soft output MIMO detector $N_r E_b/N_0 = 1.25$dB

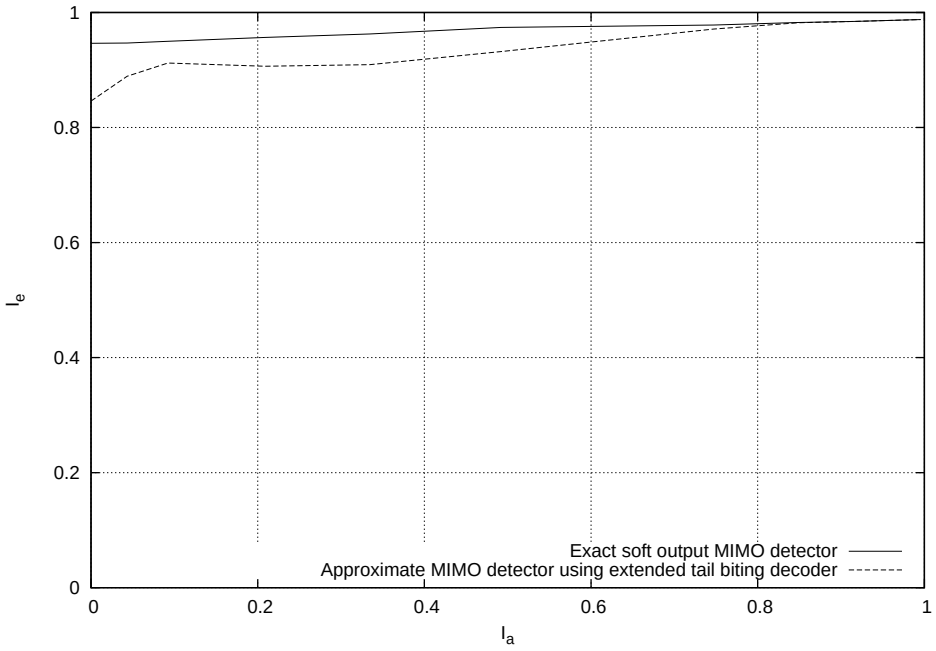

Figure 6.13: EXIT curves of the approximate MIMO detector using extended tail biting decoder and the exact soft output MIMO detector $N_r E_b/N_0 = 6.02$dB

approximate estimation of the MIMO channel capacity [36].

We computed the EXIT curves at three different SNR values. These results are shown in Figures 6.11, 6.12, and 6.13. Since the area between the two EXIT curves in Figure 6.11 is negligible, the proposed algorithm can be used in an iterative detection-decoding scheme with the same code as the optimum algorithm at low SNR or in the power limited region. The EXIT curves shown in Figure 6.12 lead to similar conclusion. The area between the two EXIT curves becomes 0.04 in Figure 6.13. This means that the proposed algorithm can also be used in the bandwidth limited region but at the cost of a rate loss of 0.04bits which is quite acceptable.

6.7 Usage of the decoders of the convolutional codes as channel equalizers

Let $X(t)$ be a stochastic process defined as follows.

$$X(t) = \sum_n \eta_N(\mathbf{X}_n) f(t - nT), \tag{6.78}$$

where $f(t)$ is the impulse response of a pulse shaping filter and $\mathbf{X}_n$ is a random vector consisting of N bits. Furthermore, let $Y(t)$ be

$$Y(t) = X(t) * g(t) + Z(t) \tag{6.79}$$

$$= \sum_n \eta_N(\mathbf{X}_n) * h(t) + Z(t), \tag{6.80}$$

where $Z(t)$ is a zero mean white Gaussian noise process with power spectral density $\frac{N_0}{2}$, $*$ denotes convolution, $g(t)$ is the impulse response of a causal channel, and $h(t)$ is the convolution of the $g(t)$ and $f(t)$. Then it can be shown by following similar procedures applied in the previous sections that the Viterbi and BCJR decoders of a certain convolutional code C can be used as ML sequence estimator and marginal APP receiver for this inter-symbol interference system respectively. This code C is the non-recursive systematic convolutional code of rate $1/NL$ and of constraint length NL where L is the smallest integer such that $h(t) = 0$ for $t > LT$. The generator polynomials of this code are $1, 1 + x, 1 + x^2, \ldots, 1 + x^{NL-1}$.

The inputs that must be applied to these decoders to achieve the desired results consists of samples taken from the output of the matched filter i.e. $y(t) * h(-t)$ with sampling period T, where $y(t)$ is the received signal, samples taken from the time autocorrelation function $h(t) * h(-t)$ again with sampling period T, and scaling of these samples with 2's powers [4].

The Viterbi decoder of the mentioned code above actually works as an alternative device to compute the Ungerboeck's metric [41]. Therefore, this result would be much more interesting if we achieved it before Ungerboeck. However, using a Viterbi decoder as an alternative device to compute Ungerboeck's might still be of practical importance since this approach takes all of the multiplications outside of the Viterbi data path.

We have also empirically verified that the BCJR decoder of the convolutional code mentioned above with the mentioned inputs returns the exact marginal APPs of the transmitted bits.

[4] We dropped conjugations since $\eta_N(\mathbf{X}_n)$ is real

CHAPTER 7

DETERMINING CONDITIONAL INDEPENDENCE RELATIONS FROM THE CANONICAL FACTORIZATION

7.1 Introduction

Investigating the conditional independence relations of random variables is important in many different disciplines [17, 21]. These conditional independence relationships are well represented by a graphical model called Markov random field (MRF) or undirected graphical model. In this section we show that the MRF representing a joint pmf can be determined from the projections of the joint PMF onto the subspaces described in Chapter 3.

This chapter begins with introducing the relation between conditional independence of two random variables and the canonical factorization. Then we explain how to determine Markov blankets from the canonical factorization. This chapter ends with comparing the canonical factorization with the Hammersley-Clifford Theorem.

7.2 Conditional Independence of Two Random Variables

Suppose that it is desired to determine the conditional independence relations between the components of the random vector $\mathbf{X} = [X_1, X_2, \ldots, X_N]$ which is distributed with a $p(\mathbf{x}) \in \mathcal{P}_{\mathbb{F}_q^N}$. Then a random variable X_i is said to be conditionally independent of X_j given all the other components of $\mathbf{X}$ if and only if the following relation is satisfied:

$$\Pr\{X_i = x_i | \mathbf{X}_{\setminus\{i\}} = \mathbf{x}_{\setminus\{i\}}\} = \Pr\left\{X_i = x_i | \mathbf{X}_{\setminus\{i,j\}} = \mathbf{x}_{\setminus\{i,j\}}\right\} \tag{7.1}$$

where $\mathbf{X}_{\setminus\mathcal{I}}$ ($\mathbf{x}_{\setminus\mathcal{I}}$) denotes the vector obtained by removing the components having indices in $\mathcal{I}$ from $\mathbf{X}$ ($\mathbf{x}$). The following theorem states the necessary and sufficient conditions for the conditional independence of two random variables in terms of the canonical factorization.

Theorem 7.1 *Let* $\mathbf{X}$ *be a random vector distributed with* $p(\mathbf{x})$ *in* $\mathcal{P}_{\mathbb{F}_q}$. X_k *and* X_l *are conditionally independent given* $\mathbf{X}_{\setminus\{k,l\}}$ *if and only if* $p(\mathbf{x})$ *can be factored as*

$$p(\mathbf{x}) = C_{\mathbb{F}_q^N}\left\{\prod_{\mathbf{a}_i\in\mathcal{K}_k\cup\mathcal{K}_l} r_i(\mathbf{a}_i\mathbf{x}^T)\right\} \tag{7.2}$$

where $\mathcal{K}_k$ *and* $\mathcal{K}_l$ *are defined as*

$$\begin{aligned}\mathcal{K}_k &\triangleq \left\{\mathbf{a}_i \in \mathcal{H} : \mathbf{f}_k\mathbf{a}_i^T = 0\right\}\\ \mathcal{K}_l &\triangleq \left\{\mathbf{a}_i \in \mathcal{H} : \mathbf{f}_l\mathbf{a}_i^T = 0\right\}.\end{aligned}$$

The proof is given Appendix A.5.1.

The forward statement of this theorem asserts that if none of the SPC factors composing the canonical factorization of $p(\mathbf{x})$ depend on both x_i and x_j simultaneously then X_i and X_j are conditionally independent given $\mathbf{X}_{\setminus\{i,j\}}$. Actually, this result is true not only for the canonical factorization but also for any factorization.

The backward statement of Theorem 7.1 states that if an SPC factor of $p(\mathbf{x})$ *with nonzero norm* depends on x_i and x_j simultaneously then X_i and X_j are definitely conditionally dependent given $\mathbf{X}_{\setminus\{i,j\}}$. On the other hand, in an ordinary factorization a factor function may depend on x_i and x_j together but X_i and X_j can still be conditionally independent given $\mathbf{X}_{\setminus\{i,j\}}$. Therefore, the backward statement of Theorem 7.1 is specific to the canonical factorization and does not hold for all factorizations in general. This fact is another reason why we call the proposed factorization the canonical factorization.

7.3 Determining Markov Blankets and the Markov Random Field

The Markov blanket of a random variable X_i, which is denoted with ∂X_i, is the *smallest* possible set containing the components of $\mathbf{X}_{\setminus\{i\}}$ which satisfies

$$\Pr\{X_i|\mathbf{X}_{\setminus\{i\}}\} = \Pr\{X_i|\partial X_i\}. \tag{7.3}$$

Clearly, ∂X_i consists of variables X_j which are not conditionally independent of X_i given $\mathbf{X}_{\setminus\{i,j\}}$. Based on Theorem 7.1, ∂X_i can be obtained in terms of projections onto the SPC constraints as follows.

Corollary 7.2 *Let the canonical factorization of $p(\mathbf{x})$ be given by*

$$p(\mathbf{x}) = C_{\mathbb{F}_q^N} \left\{ \prod_{\mathbf{a}_i \in \mathcal{H}} r_i(\mathbf{a}_i \mathbf{x}^T) \right\}. \tag{7.4}$$

X_k is in ∂X_l if and only if there exist a parity check coefficient vector $\mathbf{a}_i \in \mathcal{H}$ such that

$$\begin{aligned} \mathbf{f}_k \mathbf{a}_i^T &\neq 0 \\ \mathbf{f}_l \mathbf{a}_i^T &\neq 0 \end{aligned}$$

and

$$||r_i(x)|| > 0.$$

Proof. If such a vector $\mathbf{a}$ exist then $p(\mathbf{x})$ cannot be factored as in (7.2) and hence, X_i and X_j are conditionally dependent given $\mathbf{X}_{\setminus\{i,j\}}$ due Theorem 7.1.

If there is no such $\mathbf{a}$ then $p(\mathbf{x})$ can be factored as in (7.2), which means that X_i and X_j are conditionally independent given $\mathbf{X}_{\setminus\{i,j\}}$. ■

The MRF is an undirected graphical model representing a probability distribution where each variable is represented with a node. The node representing X_i is connected to the node representing X_j in the MRF if X_j is in ∂X_i. Since the Markov blankets of every variable can be determined from the canonical factorization by Corollary 7.2, the MRF can also be determined from the canonical factorization.

Notice that every argument (arguments associated with nonzero parity check coefficient) of a non-constant SPC factor are in the Markov blankets of the other arguments of the SPC factor. Therefore, the nodes representing these variables in the MRF are all pairwise connected. In graph theoretic terminology, these nodes form a clique in the MRF. Hence, SPC factors are functions of the cliques (not necessarily maximal) of the MRF.

7.4 Comparison to the Hammersley-Clifford Theorem

The relation between the factorization of a multivariate PMF and Markov properties is first established by Hammersley and Clifford in [18, 19]. In this work they show that any *strictly positive* multivariate PMF can be expressed as

$$p(\mathbf{x}) = \frac{1}{C} \prod_{\mathbf{D} \in \mathcal{D}_C} \phi_{\mathbf{D}}(\mathbf{D}\mathbf{x}), \tag{7.5}$$

where each element of $\mathcal{D}_C$ is associated with a clique in the MRF. Moreover, their proof is constructive. The factor functions are given as

$$\phi_{\mathbf{D}}(\mathbf{D}\mathbf{x}) \triangleq \prod_{\mathbf{D}' : \mathbf{D}'\mathbf{D} = \mathbf{D}'} p(\mathbf{D}'\mathbf{x} + (\mathbf{I} - \mathbf{D}')\mathbf{x}_B)^{\left((-1)^{|\mathbf{D}-\mathbf{D}'|}\right)} \tag{7.6}$$

where $\mathbf{x}_B$ is a fixed configuration [1]. Although both in our and their approaches the factor functions appear to be the functions of the cliques of the MRF, our approach differs significantly from theirs in many aspects.

First of all, the dependencies between the random variables imposed by factor functions in (7.6) are rather arbitrary. SPC factors, on the other hand, impose an algebraic form of dependency. In other words, SPC factors explain how a random variable is related to a linear combination of other variables. This property is quite important and allows us to express an inference problem as a decoding problem.

Second, the factor functions defined in (7.6) depend on a certain fixed configuration $\mathbf{x}_B$. A different factorization is obtained for each different $\mathbf{x}_B$. Therefore, the factorization proposed by Hammersley and Clifford is not unique. On the other hand, the canonical factorization is unique as explained in Section 4.4.

In addition, there is at most one factor function per clique in the factorization given in (7.5) whereas there may be more than one SPC factors depending on the same set of variables in non-binary fields.

Finally, the applicability of our approach is more restricted than that of the Hammersley and Clifford's. Our method is applicable only if the event space of the combined experiment can be mapped to $\mathbb{F}_q^N$ whereas the Hammersley-Clifford theorem is applicable to any strictly positive pmf. Moreover, it should be emphasized that both approaches are applicable to strictly positive pmfs only.

[1] This configuration corresponds to the all-black coloring in [18, 19].

CHAPTER 8

Conclusions and Future Directions

8.1 Summary

In this thesis the Hilbert space of pmfs is introduced. Then the tools provided by this Hilbert space, is utilized to develop an analysis method for multivariate pmfs. The aim of this analysis method is to obtain a factorization of the multivariate pmf. The resulting factorization from this analysis method possess some important properties. First of all it is the ultimate factorization possible. Secondly, it is unique. Thirdly, the conditional independence relations can be determined completely from this factorization. Probably the most important property of the resulting factorization is the fact that it reveals the algebraic dependencies between the involved random variables. Thanks to this fact probabilistic inference problems can be transformed into channel decoding problems and channel decoders can be used for other tasks beyond decoding. Many examples are provided in thesis on how channel decoders can be used as detectors of communication receivers. It is also shown that the decoders of tail biting convolutional codes can be used as a MIMO detector. This approach results in a significant reduction in complexity while maintaining good performance.

8.2 Future directions

The application of the Hilbert space of pmfs is presented in this thesis is the canonical factorization. We believe that the Hilbert space of pmfs might lead to further applications in communication theory, information theory, and probabilistic inference.

The most important consequence of the canonical factorization is that it shows how to em-

ploy channel decoders for other purposes. The MIMO detector which uses the decoder of a tail biting convolutional code demonstrates that new detection and probabilistic inference algorithms can be developed by using channel decoders for tasks beyond decoding.

Employing channel decoders for other tasks also allows to apply the analog probability propagation method proposed in [14, 15] for other probabilistic inference problems. In particular, by implementing channel equalizers and MIMO detectors with analog probability propagation much more power efficient communication receivers can be implemented. We anticipate that this direction will be the most important application area of this thesis.

Some other possible future directions are summarized below.

8.2.1 Applications on machine learning

Estimating the factorization of a joint pmf from samples generated from the pmf is an important problem in machine learning, e.g. [20]. A straightforward approach after this thesis could be estimating the joint pmf first and obtain the canonical factorization by applying the procedure explained in Chapter 3. However, such an approach both require too many samples to estimate the joint pmf accurately and extensive computational resources to obtain the canonical factorization. A more interesting solution to this problem might be proposed by combining the results obtained in this thesis and the results presented in [32]. By combining these results it can be concluded that the necessary algorithm for estimating the factorization of a joint pmf from samples is exactly the *inverse of the sum-product algorithm*.

As it is explained in Section 4.3 the ultimate factorization of a pmf is the canonical factorization. The equivalent Tanner graph representing the canonical factorization is shown in Figure 5.3-b. Hence, estimating the canonical factorization is equivalent to estimating all of the local evidences in this Tanner graph.

Let $\mathbf{X} = [X_1, X_2, \ldots, X_N]$ be distributed with a $p(\mathbf{x})$ in $\mathcal{P}_{\mathbb{F}_q^N}$. Estimating all the marginals $\Pr\{X_i = x_i\}$ from experimental data is much easier than estimating the joint distribution $p(\mathbf{x})$ from data. Let

$$X_i \triangleq \mathbf{a}_i \mathbf{X}^T, \quad \text{for} i = N+1, N+2, \ldots, |\mathcal{H}|,$$

where $\mathbf{a}_{N+1}$, $\mathbf{a}_{N+2}$, ..., $\mathbf{a}_{|\mathcal{H}|}$ are the elements of $\mathcal{H}$ of weight two or more as we assumed in Chapter 5. Since X_i for $i > N$ is completely determined by $\mathbf{X}$, the marginal distributions of X_i

for $i > N$ can also be estimated from the data. Consequently, the marginal distributions of X_1, X_2, ..., $X_{|\mathcal{H}|}$ can be easily estimated from the experimental data.

However, what we need to estimate the canonical factorization are not the marginal distributions of the random variables X_1, X_2, ..., $X_{|\mathcal{H}|}$ but the local evidences in Figure 5.3-b. Therefore, we need an algorithm which computes the local evidences from the marginals. Notice that, this task is exactly *the inverse of the sum-product algorithm* as the sum-product algorithm computes the marginals from local evidences.

A question might arise on the existence and uniqueness of the set of the local evidences corresponding to a set of marginals. Indeed, if the Tanner graph in Figure 5.3-b represented an arbitrary code then we might not find a set of local evidences resulting in a given set of marginal distributions at all or might find more than one set of local evidences resulting in the same set of marginal distributions. Any linear combination of the vector $\mathbf{X}$ is equal to αX_i for an $\alpha \in \mathbb{F}_q$ and $1 \leq i \leq |\mathcal{H}|$. Massey showed in [32] that the marginal distributions of the linear combinations of a sequence of random variables is enough to specify their joint distribution. Hence, the marginal distributions of X_1, X_2, ..., $X_{|\mathcal{H}|}$ uniquely specifies $p(\mathbf{x})$ and consequently its canonical factorization.

To the best of our knowledge, neither exact nor approximate versions of the inverse of the sum-product algorithm is known. As explained above, developing the inverse of the sum-product algorithm solves an important problem in machine learning.

8.2.2 Using channel decoders for channel estimation

In the examples presented in Chapter 6, we assumed that the channel coefficients are completely known at the receiver. In a practical communication receiver, the channel coefficients must be estimated. Employing channel decoders for channel estimation would be very interesting.

Actually, the channel estimation problem does not perfectly fit into the framework presented in this thesis since the channel coefficients take samples from a continuous alphabet rather than a finite alphabet. The apparent solution to this problem might be quantizing the channel coefficients. However, such an approach would lead to a factor graph topologically equivalent to the one in [42] which contains too many short cycles. Hence, such an approach probably

will not be useful.

While employing decoders for detection, we observed that the channel coefficients and the channel outputs appeared as the parameters of the canonical factorization of the transmitted bits. Therefore, a more interesting approach might be bypassing the channel estimation step and estimating the canonical factorization of the joint pmf of the transmitted bits and the quantized channel outputs directly from a pilot sequence. This approach transforms the channel estimation problem into a machine learning problem a solution to which is conjectured in the previous section.

REFERENCES

[1] F. R. Kschischang, B. J. Frey, and H. A. Loeliger, "Factor Graphs and the Sum-Product Algorithm", IEEE Transactions on Information Theory, vol.47, No.2, pp.498-519 February 2001

[2] H. A. Loeliger, "An Introduction to Factor Graphs", IEEE Signal Processing Magazine, Vol. 21, Issue 1, pp.28-41 Jan. 2004

[3] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon Limit error correcting coding and decoding: Turbo Codes" in Proc. 1993 IEEE Int. Conf. Comm., Gen., Switzerland, May. 1993, pp. 1064-1070

[4] J. J. Egozcue, J. L. Diaz-Barrero, V. Pawlowsky-Glahn, "A Hilbert Space of Probability Density Functions Based on Aitchison Geometry", Acta Mathematica Sinica, Springer Berlin, July 2006

[5] R. M. Tanner, "A Recursive Approach to Low-Complexity Codes", IEEE Transactions on Information Theory, vol. 27, pp. 533-547, Sept. 1981

[6] N. Wiberg, "Codes and Decoding on General Graphs", Ph. D. Thesis, Department of Electrical Eng. Linköping University, Linköping, Sweden, 1996

[7] N. Wiberg, H.-A. Loeliger, and R. Kötter, "Codes and Iterative Decoding on General Graphs", European Transactions on Communications, vol. 6, pp. 513-525, Sept./Oct. 1995

[8] G. David Forney Jr., "Codes on Graphs: Normal Realizations", IEEE Transactions on Information Theory, vol. 27, pp. 520-548, February 2001

[9] M. F. Bayramoğlu and A. Ö. Yılmaz, "A Hilbert Space of Probability Mass Functions and Applications on the Sum-Product Algorithm", Proc. 5^{th} Int. Symp. On Turbo Codes, pp.338-343, Lausanne, Sept. 2008

[10] L. Barnault and D. Declercq, "Fast Decoding Algorithms for LDPC over GF(2^q)", Proc. ITW2003, pp.70-73, Paris, April 2003

[11] Richard E. Blahut, "Algebraic Codes for Data Transmission", Cambridge Univ. Press 2003

[12] Charles A. Desoer, " Notes for a Second Course on Linear Systems", Van Nostrand Reinhold, New York, 1970

[13] I. Land and J. Huber, "Information Combining", Foundations and Trends in Information Theory, pp. 227-330, 2006

[14] H.-A. Loeliger, F. Lustenberger, M. Helfenstein, and F. Tarkoy, "Probability Propagation and Decoding in Analog VLSI", IEEE Tran. on Information Theory, pp.837-843, February 2001

[15] Hans-Andrea Loeliger, *"Analog Decoding and Beyond"*, Information Theory Workshop, Cairns, Australia, September 2001

[16] M. M. Mansour and N. R. Shanbhag, "Low-Power VLSI Decoder Architectures for LDPC Codes", Proc. ISLPED 2002

[17] Christopher M. Bishop, "Pattern Recognition and Machine Learning", Springer, New York 2006

[18] J. M. Hammersley and P. Clifford "Markov fields on finite graphs and lattices". Unpublished, 1971.

[19] Peter Clifford, "Markov Random Fields in Statistics", Disorder in Physical Systems, pp.19-32, Oxford University Press, 1990

[20] P. Abbeel, D. Koller, A. Y. Ng, "Learning Factor Graphs in Polynomial Time and Sample Complexity", Journal of Machine Learning Research vol.7 1743-1788, 2006

[21] R. Kindermann, J. L. Snell, "Markov Random Fields and Their Applications ", American Mathematical Society, Rhode Island, 1980

[22] John G. Proakis, "Digital Communications", McGraw Hill, 2001

[23] T. S. Blyth and E. F. Robertson, "Further Linear Algebra", Springer, 2002

[24] L. R. Bahl, J. Cocke, F. Jelinek, and J.Raviv, "Optimal decoding of linear codes for minimizing symbol error rate", IEEE Transactions on Information Theory, vol. IT-20, pp.284-287, Mar. 1974

[25] Harry L. van Trees, "Detection, Estimation, and Modulation Theory", John Wiley&Sons, 2001

[26] Simon Haykin, "Communication Systems", John Wiley&Sons, 1994

[27] A. G. Dabak. "A Geometry for Detection Theory". PhD thesis, Dept. Electrical and Computer Engineering, Rice University, Houston, TX, 1992.

[28] T. P. Minka, "A family of algorithms for approximate Bayesian inference", Ph. D. Thesis, Department of Electrical Eng. and Computer Science, Massachusetts Institute of Technology, 2001

[29] J. Hu, H.-A. Loeliger, J. Dauwels, and F. Kschischang, " A general computation rule for lossy summaries/messages with examples from equalization" Proc. 44th Allerton Conf. on Communication, Control, and Computing, Monticello, Illinois, Sept. 27-29, 2006.

[30] H. L. van Trees, "Detection, Estimation, and Modulation Theory", John Wiley&Sons, 2001

[31] A. Papoulis, "Probability, Random Variables and Stochastic Processes" McGraw Hill, 1991

[32] James L. Massey, "Randomness, Arrays, Differences and Duality", IEEE Tran. on Information Theory, vol. 48, pp. 1698-1703, June 2002

[33] David J. C. MacKay, "Information Theory, Inference, and Learning Algorithms", Cambridge University Press, 2003

[34] Carver Mead, “Analog VLSI and Neural Systems”, Addison-Wesley ,1989

[35] Stephan ten Brink, “Convergence Behavior of Iteratively Decoded Parallel Concatenated Codes”, IEEE Tran. on Communications, Vol. 49, No. 10, pp. 1727-1737, October 2001

[36] A. Ashikhmin, G. Kramer, and S. ten Brink, “Extrinsic Information Transfer Functions: Model and Erasure Channel Properties”, IEEE Tran. on Information Theory, Vol. 50, No. 11, pp. 2657-2673, November 2004

[37] S. ten Brink, G. Kramer, and A. Ashikhmin, “Design of Low-Density Parity-Check Codes for Modulation and Detection”, IEEE Tran. on Communications, vol. 52, pp. 670-678, April 2004

[38] M. E. O’Sullivan, J. Brevik, and S. M. Vargo, “The Sum-Product Algorithm on Simple Graphs”, Information Theory and Applications Workshop, San Diego, CA, USA, February 2009

[39] S. M. Aji, G. B. Horn, and R. J. McEliece, “Iterative Decoding on Graphs with a Single Cycle”, Proceedings of the International Symposium on Information Theory, Cambridge, MA, USA, August 1998

[40] K. P. Murphy, Y. Weiss, M. I. Jordan, “Loopy Belief Propagation for Approximate Inference: An Empirical Study”, Proceedings of the Uncertainty in AI, 1999

[41] Gottfried Ungerboeck, “Adaptive Maximum-Likelihood Receiver for Carrier-Modulated Data Transmission Systems”, IEEE Tran. on Communications, Vol 22, No. 5, pp. 624-636, May 1974

[42] A. P. Worthen and W. E. Stark, “Unified Design of Iterative Receivers Using Factor Graphs”, IEEE Tran. on Information Theory, vol. 47, no. 2, pp. 843-849, Feb. 2001

[43] E. Biglieri, R. Calderbank, A. Constantinides, A. Goldsmith, A. Paulraj, and H. V. Poor, “MIMO Wireless Communications”, Cambridge University Press, 2007

APPENDIX A

PROOFS AND DERIVATIONS

A.1 Proofs and derivations in Chapter 2

A.1.1 Proof of Lemma 2.2

The function $\sigma(p(x), r(x))$ defined in (2.19) is an inner product on $\mathcal{P}_{\mathbb{F}_q}$ if it satisfies three inner product axioms stated below.

- *Symmetry:* This property of $\sigma(.,.)$ is directly inherited from the inner product on $\mathbb{R}^q$.

- *Linearity w.r.t. first argument:* If $\mathcal{M}\{.\}$ is linear this property is also inherited from the inner product on $\mathbb{R}^q$.

- *Positive definiteness:* For any $p(x) \in \mathcal{P}_{\mathbb{F}_q}$

$$\begin{aligned} \sigma(p(x), p(x)) &= < \mathcal{M}\{p(x)\}, \mathcal{M}\{p(x)\} > \\ &\geq 0 \end{aligned}$$

  due to the non-negativity of the inner product on $\mathbb{R}^q$. The equality is satisfied only if $\mathcal{M}\{p(x)\}$ equals to $\mathbf{0}$. Since $\mathcal{M}\{.\}$ is linear and an injection $\mathcal{M}\{p(x)\}$ is equal to $\mathbf{0}$ if and only if $p(x) = \theta(x)$.

A.1.2 Rationale behind the proposal for $\mathcal{L}\{.\}$

The trivial way of mapping a pmf $p(x) \in \mathcal{P}_{\mathbb{F}_q}$ by a vector $\mathbf{p} \in \mathbb{R}^q$ is making the i^{th} [1] component of $\mathbf{p}$ equal to $p(i)$. Let this trivial mapping be denoted by $\mathcal{T}\{.\}$, i.e.,

$$\mathcal{T}\{p(x)\} \triangleq \sum_{i \in \mathbb{F}_q} p(i)\mathbf{e}_i.$$

Although this mapping is injective, it is obviously nonlinear. Therefore, $\mathcal{T}\{.\}$ does not satisfy one of the two requirements imposed by Lemma 2.2 and consequently it cannot be employed as a tool for borrowing the inner product on $\mathbb{R}^q$. However, we can define a notion of angle between pmfs using $\mathcal{T}\{.\}$ and then reach a proposal for a mapping which satisfies the requirements of Lemma 2.2.

Whatever the definition of the angle between two pmfs is, the sine of the angle should be kept constant if two pmfs are scaled by some nonzero scalars. In other words, for any $p(x), r(x) \in \mathcal{P}_{\mathbb{F}_q}$ and $\alpha, \beta \in \mathbb{R} \setminus \{0\}$

$$\sin \angle(p(x), r(x)) = \sin \angle(\alpha \boxtimes p(x), \beta \boxtimes r(x)),$$

where $\angle(p(x), r(x))$ denotes the angle between $p(x)$ and $r(x)$. This property of angle imposes that the angle between two pmfs should be a function of the two parametric curves on $\mathbb{R}^q$ based on $p(x)$ and $r(x)$ as follows.

$$\begin{aligned} \mathbf{c}_p(t) &\triangleq \mathcal{T}\{t \boxtimes p(x)\}, \\ \mathbf{c}_r(t) &\triangleq \mathcal{T}\{t \boxtimes r(x)\}. \end{aligned}$$

For $t = 0$ both of these curves pass through $\frac{1}{q}\mathbf{1}$. An example consisting of a pair of such curves for $\mathcal{P}_{\mathbb{F}_3}$ is depicted in Figure A.1. Then we can reasonably define the angle between $p(x)$ and $r(x)$ as the angle between $\mathbf{c}_p(t)$ and $\mathbf{c}_r(t)$ at their intersection point.

In order to derive the angle between $\mathbf{c}_p(t)$ and $\mathbf{c}_r(t)$, we need to derive vectors tangent to these curves at $t = 0$. The expression defining $\mathbf{c}_p(t)$ can be simplified as

$$\begin{aligned} \mathbf{c}_p(t) &= \sum_{i \in \mathbb{F}_q} \frac{(p(i))^t}{\sum_{j \in \mathbb{F}_q} (p(j))^t} \mathbf{e}_i \\ &= \sum_{i \in \mathbb{F}_q} \left(\sum_{j \in \mathbb{F}_q} \exp\Big(t\big(\log p(j) - \log p(i)\big)\Big) \right)^{-1} \mathbf{e}_i. \end{aligned}$$

[1] We enumerate the components of the vector with the elements of $\mathbb{F}_q$ instead of positive integers.

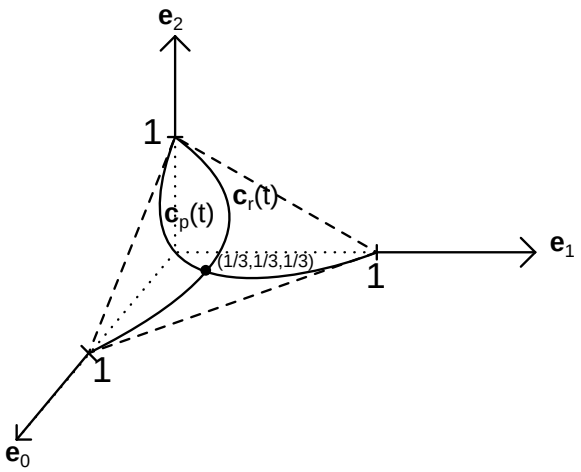

Figure A.1: A pair of parametric curves obtained by scaling two pmfs in $\mathcal{P}_{\mathbb{F}_3}$ and then mapping them to $\mathbb{R}^3$ via the trivial mapping.

Let $\mathbf{t}_p$ denote the vector which is tangent to $\mathbf{c}_p(t)$ at $t = 0$. Then $\mathbf{t}_p$ can be derived using derivation as

$$\mathbf{t}_p = \sum_{i \in \mathbb{F}_q} \left(q \log p(i) - \sum_{j \in \mathbb{F}_q} \log p(j) \right) \mathbf{e}_i.$$

Having inspired from this equation, We proposed the mapping $\mathcal{L}\{.\}$ as

$$\begin{aligned} \mathcal{L}\{.\} &= \frac{1}{q} \mathbf{t}_p \\ &= \sum_{i \in \mathbb{F}_q} \left(\log p(i) - \frac{1}{q} \sum_{j \in \mathbb{F}_q} \log p(j) \right) \mathbf{e}_i. \end{aligned}$$

Since $\mathcal{L}\{.\}$ is defined as above, the angle between the two curves $\mathbf{c}_p(t)$ and $\mathbf{c}_q(t)$, which is proposed to be the of the angle between $p(x)$ and $q(x)$, is equal to the angle between $p(x)$ and $q(x)$ on $\mathcal{P}_{\mathbb{F}_q}$ defined on (2.28).

A.1.3 Proof of Lemma 2.3

First we are going to prove that $\mathcal{L}\{.\}$ is linear and then it is an injection. For any $p(x), r(x) \in \mathcal{P}_{\mathbb{F}_q}$,

$$\begin{aligned}
\mathcal{L}\{p(x) \boxplus r(x)\} &= \sum_{i\in\mathbb{F}_q}\left(\log C_{\mathbb{F}_q}\{p(x)r(x)\}\Big|_{x=i} - \frac{1}{q}\sum_{j\in\mathbb{F}_q}\log C_{\mathbb{F}_q}\{p(x)r(x)\}\Big|_{x=j}\right)\mathbf{e}_i \\
&= \sum_{i\in\mathbb{F}_q}\left(\log\frac{1}{\gamma}p(i)r(i) - \frac{1}{q}\sum_{j\in\mathbb{F}_q}\log\frac{1}{\gamma}p(j)r(j)\right)\mathbf{e}_i \\
&= \sum_{i\in\mathbb{F}_q}\left(\log p(i) - \frac{1}{q}\sum_{j\in\mathbb{F}_q}\log p(j)\right)\mathbf{e}_i + \sum_{i\in\mathbb{F}_q}\left(\log p(i) - \frac{1}{q}\sum_{j\in\mathbb{F}_q}\log p(j)\right)\mathbf{e}_i \\
&= \mathcal{L}\{p(x)\} + \mathcal{L}\{r(x)\},
\end{aligned}$$

where γ in the second line above is $\sum_{i\in\mathbb{F}_q} p(i)r(i)$. Hence, $\mathcal{L}\{.\}$ is additive. For any $p(x) \in \mathcal{P}_{\mathbb{F}_q}$ and $\alpha \in \mathbb{R}$,

$$\begin{aligned}
\mathcal{L}\{\alpha \boxtimes p(x)\} &= \sum_{i\in\mathbb{F}_q}\left(\log C_{\mathbb{F}_q}\{(p(x))^\alpha\}\Big|_{x=i} - \frac{1}{q}\sum_{j\in\mathbb{F}_q}\log C_{\mathbb{F}_q}\{(p(x)^\alpha)\}\Big|_{x=j}\right)\mathbf{e}_i \\
&= \sum_{i\in\mathbb{F}_q}\alpha\left(\log p(i) - \frac{1}{q}\sum_{j\in\mathbb{F}_q}\log p(j)\right)\mathbf{e}_i \\
&= \alpha\mathcal{L}\{p(x)\}.
\end{aligned}$$

Hence, $\mathcal{L}\{.\}$ is homogeneous and consequently a linear mapping.

A linear mapping is injective if its kernel (null space) is composed of only the additive identity. If $\mathcal{L}\{p(x)\} = \mathbf{0}$ for a $p(x) \in \mathbb{F}_q$ then

$$\log p(i) - \frac{1}{q}\sum_{j\in\mathbb{F}_q}\log p(j) = 0 \qquad \forall i \in \mathbb{F}_q$$

$$p(i) = \exp\left(\frac{1}{q}\sum_{j\in\mathbb{F}_q}\log p(j)\right) \qquad \forall i \in \mathbb{F}_q,$$

which is possible only if $p(x) = \frac{1}{q}$ or equivalently $p(x) = \theta(x)$. Since the kernel of $\mathcal{L}\{.\}$ consists of only $\theta(x)$, which is the additive identity in $\mathcal{P}_{\mathbb{F}_q}$, the mapping $\mathcal{L}\{.\}$ is injective.

A.1.4 Expressing the inner product on $\mathcal{P}_{\mathbb{F}_q}$ as a covariance

Let X be a $\mathbb{F}_q$-valued random variable. Then $\log p(X)$ and $\log r(X)$ are two real-valued functions of an $\mathbb{F}_q$-valued random variable. Their expectations and covariance are well-defined.

Clearly, the inner product of $p(x)$ and $r(x)$ can be expressed as

$$\begin{aligned} < p(x), r(x) > &= q\mathbf{E}\left[(\log p(X) - \mathbf{E}\left[\log p(X)\right])(\log r(X) - \mathbf{E}\left[\log r(X)\right])\right] \\ &= q\left(\mathbf{E}\left[\log p(X)\log r(X)\right] - \mathbf{E}\left[\log p(X)\right]\mathbf{E}\left[\log r(X)\right]\right), \end{aligned}$$

where $\mathbf{E}\left[.\right]$ denotes expectation and X is a uniformly distributed random variable in $\mathbb{F}_q$, i.e.

$$\Pr\{X = x\} = \theta(x).$$

A.1.5 Proof of Lemma 2.5

For any $p(x)$ in $\mathcal{P}_{\mathbb{F}_q}$

$$\begin{aligned} < \mathcal{L}\left\{p(x)\right\}, \mathbf{1} >_{\mathbb{R}^q} &= < \sum_{i\in\mathbb{F}_q}\left(\log p(i) - \frac{1}{q}\sum_{j\in\mathbb{F}_q}\log p(j)\right)\mathbf{e}_i, \mathbf{1} >_{\mathbb{R}^q} \\ &= \sum_{i\in\mathbb{F}_q}\left(log p(i) - \frac{1}{q}\sum_{j\in\mathbb{F}_q}\log p(j)\right) \\ &= \sum_{i\in\mathbb{F}_q}\log p(i) - \sum_{j\in\mathbb{F}_q}\log p(j) \\ &= 0, \end{aligned}$$

which completes the proof.

A.1.6 Proof of Lemma 2.6

First we are going to simplify the expression defining $\mathcal{L}^+\left\{\mathbf{p}\right\}(x)$.

$$\begin{aligned} \mathcal{L}^+\left\{\mathbf{p}\right\}(x) &= C_{\mathbb{F}_q}\left\{\exp\left(-\frac{1}{2}\left\|\mathbf{p} - \mathbf{s}(x)\right\|^2\right)\right\} \\ &= C_{\mathbb{F}_q}\left\{\exp\left(-\frac{\|\mathbf{p}\|^2 - 2 < \mathbf{p}, \mathbf{s}(x) >_{\mathbb{R}^q} + \|\mathbf{s}(x)\|^2}{2}\right)\right\} \\ &= C_{\mathbb{F}_q}\left\{\exp\left(-\frac{\|\mathbf{p}\|^2}{2}\right)\exp\left(-\frac{\|\mathbf{s}(x)\|^2}{2}\right)\exp\left(< \mathbf{p}, \mathbf{s}(x) >_{\mathbb{R}^q}\right)\right\} \end{aligned}$$

Since $\|\mathbf{p}\|$ and $\|\mathbf{s}(x)\|$ is constant for all x, the product $\exp\left(-\frac{\|\mathbf{p}\|^2}{2}\right)\exp\left(-\frac{\|\mathbf{s}(x)\|^2}{2}\right)$ has no effect due to the normalization operator. Therefore,

$$\mathcal{L}^+\left\{\mathbf{p}\right\}(x) = C_{\mathbb{F}_q}\left\{\exp\left(< \mathbf{p}, \mathbf{s}(x) >_{\mathbb{R}^q}\right)\right\}. \tag{A.1}$$

If $\mathbf{p}$ is equal to $\mathcal{L}\{p(x)\}$ for a $p(x)$ in $\mathcal{P}_{\mathbb{F}_q}$ then the inner product above becomes

$$\begin{aligned} < \mathbf{p}, \mathbf{s}(x) >_{\mathbb{R}^q} &= < \mathcal{L}\{p(x)\}, \mathbf{e}_x - \frac{1}{q}\mathbf{1} > \\ &= < \mathcal{L}\{p(x)\}, \mathbf{e}_x > - \frac{1}{q} < \mathcal{L}\{p(x)\}, \mathbf{1} > . \end{aligned}$$

Due to Lemma 2.5 the second inner product above is zero. Inserting this result into (A.1) yields

$$\begin{aligned} \mathcal{L}^+\{\mathcal{L}\{p(x)\}\}(x) &= C_{\mathbb{F}_q}\left\{\exp\left(\log p(x) - \frac{1}{q}\sum_{j\in\mathbb{F}_q}\log p(j)\right)\right\} \\ &= C_{\mathbb{F}_q}\{\exp(\log p(x))\} \\ &= p(x), \end{aligned}$$

where the summation in the first line above is cancelled by the normalization operator since it is a constant. This completes the proof of the first part of the lemma.

Any $\mathbf{p} \in \mathbb{R}^q$ can be decomposed as

$$\mathbf{p} = \mathbf{p}' + \alpha\mathbf{1},$$

for an α in $\mathbb{R}$ such that $\mathbf{p}' \perp \mathbf{1}$. Inserting this decomposition into (A.1) yields

$$\begin{aligned} \mathcal{L}^+\{\mathbf{p}\}(x) &= C_{\mathbb{F}_q}\{\exp(< \mathbf{p}' + \alpha\mathbf{1}, \mathbf{s}(x) >_{\mathbb{R}^q})\} \\ &= C_{\mathbb{F}_q}\{\exp(< \mathbf{p}', \mathbf{s}(x) >_{\mathbb{R}^q} + \alpha < \mathbf{1}, \mathbf{s}(x) >_{\mathbb{R}^q})\} \end{aligned}$$

$\mathbf{s}(x)$ is orthogonal to $\mathbf{1}$ for all x. Therefore,

$$\begin{aligned} \mathcal{L}^+\{\mathbf{p}\}(x) &= C_{\mathbb{F}_q}\{\exp(< \mathbf{p}', \mathbf{s}(x) >_{\mathbb{R}^q})\} \\ \mathcal{L}\{\mathcal{L}^+\{\mathbf{p}\}(x)\} &= \sum_{i\in\mathbb{F}_q}\left(\log\frac{1}{\gamma}\exp(< \mathbf{p}', \mathbf{s}(i) >_{\mathbb{R}^q}) - \frac{1}{q}\sum_{j\in\mathbb{F}_q}\log\frac{1}{\gamma}\exp(< \mathbf{p}', \mathbf{s}(j) >_{\mathbb{R}^q})\right)\mathbf{e}_i \\ &= \sum_{i\in\mathbb{F}_q}\left(< \mathbf{p}', \mathbf{s}(i) >_{\mathbb{R}^q} - \frac{1}{q} < \mathbf{p}', \sum_{j\in\mathbb{F}_q}\mathbf{s}(j) >_{\mathbb{R}_q}\right)\mathbf{e}_i, \end{aligned}$$

where $\gamma = \sum_{i\in\mathbb{F}_q}\exp(< \mathbf{p}', \mathbf{s}(i) >_{\mathbb{R}^q})$. $\sum_{j\in\mathbb{F}_q}\mathbf{s}(j)$ is equal to the zero vector. Therefore,

$$\begin{aligned} \mathcal{L}\{\mathcal{L}^+\{\mathbf{p}\}(x)\} &= \sum_{i\in\mathbb{F}_q}(< \mathbf{p}', \mathbf{s}(i) >_{\mathbb{R}^q})\,\mathbf{e}_i \\ &= \sum_{i\in\mathbb{F}_q}\left(< \mathbf{p}', \mathbf{e}_i + \frac{1}{q}\mathbf{1} >_{\mathbb{R}^q}\right)\mathbf{e}_i \\ &= \sum_{i\in\mathbb{F}_q}\left(< \mathbf{p}', \mathbf{e}_i >_{\mathbb{R}^q} + \frac{1}{q} < \mathbf{p}', \mathbf{1} >_{\mathbb{R}^q}\right)\mathbf{e}_i \\ &= \mathbf{p}' \end{aligned}$$

If $\mathbf{p}$ is orthogonal to $\mathbf{1}$ then $\mathbf{p}$ becomes equal to $\mathbf{p}'$ and consequently

$$\mathcal{L}\left\{\mathcal{L}^{+}\left\{\mathbf{p}\right\}(x)\right\} = \mathbf{p}.$$

A.2 Proofs and derivations in Chapter 3

A.2.1 Proof of Lemma 3.1

Inserting the expressions for $p_1(\mathbf{x})$ and $p_2(\mathbf{x})$ into inner product definition yields

$$\begin{aligned}
< p_1(\mathbf{x}), p_2(\mathbf{x}) > &= \sum_{\mathbf{i}\in\mathbb{F}_q^N} \log \frac{r_1(\mathbf{a}\mathbf{i}^T)}{q^{N-1}} \log \frac{r_2(\mathbf{b}\mathbf{i}^T)}{q^{N-1}} - \frac{1}{q^N} \sum_{\mathbf{i}\in\mathbb{F}_q^N} \log \frac{r_1(\mathbf{a}\mathbf{i}^T)}{q^{N-1}} \sum_{\mathbf{i}\in\mathbb{F}_q^N} \log \frac{r_2(\mathbf{b}\mathbf{i}^T)}{q^{N-1}} \\
&= \sum_{\mathbf{i}\in\mathbb{F}_q^N} \log r_1(\mathbf{a}\mathbf{i}^T) \log \frac{r_2(\mathbf{b}\mathbf{i}^T)}{q^{N-1}} - \frac{1}{q^N} \sum_{\mathbf{i}\in\mathbb{F}_q^N} \log r_1(\mathbf{a}\mathbf{i}^T) \sum_{\mathbf{i}\in\mathbb{F}_q^N} \log \frac{r_2(\mathbf{b}\mathbf{i}^T)}{q^{N-1}} \\
&= \sum_{\mathbf{i}\in\mathbb{F}_q^N} \log r_1(\mathbf{a}\mathbf{i}^T) \log r_2(\mathbf{b}\mathbf{i}^T) - \frac{1}{q^N} \sum_{\mathbf{i}\in\mathbb{F}_q^N} \log r_1(\mathbf{a}\mathbf{i}^T) \sum_{\mathbf{i}\in\mathbb{F}_q^N} \log r_2(\mathbf{b}\mathbf{i}^T) \quad \text{(A.2)}
\end{aligned}$$

First we are going to derive the inner product of the two SPC constraints if there exist an $\alpha \in \mathbb{F}_q$ such that $\mathbf{b} = \alpha\mathbf{a}$. Since $\mathbb{F}_q$ is a field with q elements and $\mathbf{a}$ is nonzero there are q^{N-1} $\mathbf{i}$ vectors satisfying the equation $\mathbf{a}\mathbf{i}^T = j$ for all $j \in \mathbb{F}_q$. Hence,

$$\begin{aligned}
< p_1(\mathbf{x}), p_2(\mathbf{x}) > &= q^{N-1} \sum_{j\in\mathbb{F}_q} \log r_1(j) \log r_2(\alpha j) - q^{N-2} \left(\sum_{j\in\mathbb{F}_q} \log r_1(j)\right)\left(\sum_{i\in\mathbb{F}_q} \log r_2(\alpha j)\right) \\
&= q^{N-1} < r_1(x), r_2(\alpha x) > ,
\end{aligned}$$

which completes the proof for the first part.

In the second part, we derive the inner product of $p_1(\mathbf{x})$ and $p_2(\mathbf{x})$ when there is not any $\alpha \in \mathbb{F}_q$ such that $\mathbf{b} = \alpha\mathbf{a}$. In other words, $\mathbf{a}$ and $\mathbf{b}$ are linearly independent. The first summation in (A.2) can be regrouped for this case as follows.

$$\begin{aligned}
\sum_{\mathbf{i}\in\mathbb{F}_q^N} \log r_1(\mathbf{a}\mathbf{i}^T) \log r_2(\mathbf{b}\mathbf{i}^T) &= \sum_{j\in\mathbb{F}_q} \left(\sum_{\mathbf{i}:\mathbf{a}\mathbf{i}^T=j} \log r_1(j) \log r_2(\mathbf{b}\mathbf{i}^T) \right) \\
&= \sum_{j\in\mathbb{F}_q} \log r_1(j) \sum_{\mathbf{i}:\mathbf{a}\mathbf{i}^T=j} \log r_2(\mathbf{b}\mathbf{i}^T) \\
&= \sum_{j\in\mathbb{F}_q} \log r_1(j) \sum_{j\in\mathbb{F}_q} \left(\sum_{\mathbf{i}:(\mathbf{a}\mathbf{i}^T=j \wedge \mathbf{b}\mathbf{i}^T=k)} \log r_2(k) \right) \\
&= \sum_{j\in\mathbb{F}_q} \log r_1(j) \sum_{j\in\mathbb{F}_q} \log r_2(k) \left(\sum_{\mathbf{i}:(\mathbf{a}\mathbf{i}^T=j \wedge \mathbf{b}\mathbf{i}^T=k)} 1 \right).
\end{aligned}$$

Since $\mathbb{F}_q$ is a field with q elements and $\mathbf{a}$, $\mathbf{b}$ are linearly independent the innermost summation above runs q^{N-2} times for all j and k. Therefore,

$$\begin{aligned}\sum_{\mathbf{i}\in\mathbb{F}_q^N} \log r_1(\mathbf{a}\mathbf{i}^T)\log r_2(\mathbf{b}\mathbf{i}^T) &= q^{N-2}\sum_{j\in\mathbb{F}_q}\log r_1(j)\sum_{k\in\mathbb{F}_q}\log r_2(k)\\ &= q^{N-2}\left(\sum_{j\in\mathbb{F}_q}\log r_1(j)\right)\left(\sum_{j\in\mathbb{F}_q}\log r_2(j)\right).\end{aligned}$$

Inserting this result into (A.2) yields

$$\begin{aligned}< p_1(\mathbf{x}), p_2(\mathbf{x}) > &= q^{N-2}\sum_{j\in\mathbb{F}_q}\log r_1(j)\sum_{j\in\mathbb{F}_q}\log r_2(j) - \frac{1}{q^N}\sum_{\mathbf{i}\in\mathbb{F}_q^N}\log r_1(\mathbf{a}\mathbf{i}^T)\sum_{\mathbf{i}\in\mathbb{F}_q^N}\log r_2(\mathbf{b}\mathbf{i}^T)\\ &= q^{N-2}\sum_{j\in\mathbb{F}_q}\log r_1(j)\sum_{j\in\mathbb{F}_q}\log r_2(j) -\\ &\quad \frac{1}{q^N}\left(\sum_{j\in\mathbb{F}_q}\log r_1(j)\sum_{\mathbf{i}:\mathbf{a}\mathbf{i}^T=j}1\right)\left(\sum_{j\in\mathbb{F}_q}\log r_2(j)\sum_{\mathbf{i}:\mathbf{b}\mathbf{i}^T=j}1\right)\\ &= q^{N-2}\sum_{j\in\mathbb{F}_q}\log r_1(j)\sum_{j\in\mathbb{F}_q}\log r_2(j) - q^{N-2}\sum_{j\in\mathbb{F}_q}\log r_1(j)\sum_{j\in\mathbb{F}_q}\log r_2(j)\\ &= 0,\end{aligned}$$

which completes the proof of the second part.

A.2.2 Proof of Lemma 3.2

First we are going to prove that $\mathrm{im}\{\mathcal{S}_a\}$ is a subspace of $\mathcal{P}_{\mathbb{F}_q^N}$ by showing that $\mathcal{S}_a\{.\}$ is a linear mapping from $\mathcal{P}_{\mathbb{F}_q}$ to $\mathcal{P}_{\mathbb{F}_q^N}$. For any $p(x), r(x)\in\mathcal{P}_{\mathbb{F}_q}$ and $\alpha,\beta\in\mathbb{R}$

$$\mathcal{S}_{\mathbf{a}}\{\alpha\boxtimes p(x)\boxplus\beta\boxtimes r(x)\} = C_{\mathbb{F}_q^N}\left\{C_{\mathbb{F}_q}\left\{(p(x))^\alpha(r(x))^\beta\right\}\Big|_{x=\mathbf{a}\mathbf{x}^T}\right\}.$$

The inner normalization operator above can be cancelled since there is another normalization outside.

$$\mathcal{S}_{\mathbf{a}}\{\alpha\boxtimes p(x)\boxplus\beta\boxtimes r(x)\} = C_{\mathbb{F}_q^N}\left\{\left((p(\mathbf{a}\mathbf{x}^T))^\alpha(r(\mathbf{a}\mathbf{x}^T))^\beta\right)\right\}.$$

Using the definition of addition and scalar multiplication on $\mathcal{P}_{\mathbb{F}_q^N}$ we obtain

$$\begin{aligned}\mathcal{S}_{\mathbf{a}}\{\alpha\boxtimes p(x)\boxplus\beta\boxtimes r(x)\} &= \alpha\boxtimes C_{\mathbb{F}_q^N}\left\{p(\mathbf{a}\mathbf{x}^T)\right\}\boxplus\beta\boxtimes C_{\mathbb{F}_q^N}\left\{r(\mathbf{a}\mathbf{x}^T)\right\}\\ &= \alpha\boxtimes\mathcal{S}_{\mathbf{a}}\{p(x)\}\boxplus\beta\boxtimes\mathcal{S}_{\mathbf{a}}\{r(x)\},\end{aligned}$$

which proves that $\mathcal{S}_{\mathbf{a}}\{.\}$ is a linear mapping. Since the image of any linear mapping is a subspace of the co-domain, $\mathrm{im}\{\mathcal{S}_{\mathbf{a}}\}$ is a subspace of $\mathcal{P}_{\mathbb{F}_q^N}$.

Obviously, $\mathcal{S}_{\mathbf{a}}\{.\}$ is an injective mapping for nonzero $\mathbf{a}$. Therefore,

$$\begin{aligned}\dim \operatorname{im}\{\mathcal{S}_{\mathbf{a}}\} &= \dim \mathcal{P}_{\mathbb{F}_q} \\ &= q-1,\end{aligned}$$

which completes the proof.

A.2.3 Proof of Lemma 3.3

If there exist an $\alpha \in \mathbb{F}_q$ such that $\mathbf{b} = \alpha\mathbf{a}$ then for any $p(x) \in \mathcal{P}_{\mathbb{F}_q}$

$$\begin{aligned}\mathcal{S}_{\mathbf{b}}\{p(x)\} &= C_{\mathbb{F}_q^N}\left\{p(\mathbf{b}\mathbf{x}^T)\right\} \\ &= C_{\mathbb{F}_q^N}\left\{p(\alpha\mathbf{a}\mathbf{x}^T)\right\} \\ &= \mathcal{S}_{\mathbf{a}}\{p(\alpha x)\}.\end{aligned}$$

Since $p(x)$ is in $\mathcal{P}_{\mathbb{F}_q}$, $p(\alpha x)$ is also in $\mathcal{P}_{\mathbb{F}_q}$. Therefore, $\operatorname{im}\{\mathcal{S}_{\mathbf{b}}\} \subset \operatorname{im}\{\mathcal{S}_{\mathbf{a}}\}$. Similarly, it can be shown that $\operatorname{im}\{\mathcal{S}_{\mathbf{a}}\} \subset \operatorname{im}\{\mathcal{S}_{\mathbf{b}}\}$. Consequently,

$$\operatorname{im}\{\mathcal{S}_{\mathbf{a}}\} = \operatorname{im}\{\mathcal{S}_{\mathbf{b}}\}$$

if $\mathbf{b}$ is equal to $\alpha\mathbf{a}$ for an $\alpha \in \mathbb{F}_q$.

If there is not any α such that $\mathbf{b} = \alpha\mathbf{a}$ then for any $p_1(\mathbf{x}) \in \operatorname{im}\{\mathcal{S}_{\mathbf{a}}\}$ and $p_2(\mathbf{x}) \in \operatorname{im}\{\mathcal{S}_{\mathbf{b}}\}$

$$< p_1(\mathbf{x}), p_2(\mathbf{x}) >= 0$$

due to Lemma 3.1. Hence,

$$\operatorname{im}\{\mathcal{S}_{\mathbf{a}}\} \perp \operatorname{im}\{\mathcal{S}_{\mathbf{b}}\}$$

if there is not any $\alpha \in \mathbb{F}_q$ such that $\mathbf{b} = \alpha\mathbf{a}$.

A.3 Proofs and Derivations in Chapter 4

A.3.1 Proof of Lemma 4.1

We need to show that $p(\mathbf{x})$ is orthogonal to $C_{\mathbb{F}_q^N}\left\{r(\mathbf{a}\mathbf{x}^T)\right\}$ for any $r(x) \in \mathcal{P}_{\mathbb{F}_q}$.

$$\begin{aligned}< p(\mathbf{x}), C_{\mathbb{F}_q^N}\left\{r(\mathbf{a}\mathbf{x}^T)\right\} > &= \sum_{\mathbf{i}\in\mathbb{F}_q} \log p(\mathbf{i}\mathbf{D}) \log \frac{r(\mathbf{a}\mathbf{i}^T)}{q^{N-1}} - \frac{1}{q^N}\sum_{\mathbf{i}\in\mathbb{F}_q} \log p(\mathbf{i}\mathbf{D}) \sum_{\mathbf{i}\in\mathbb{F}_q} \log \frac{r(\mathbf{a}\mathbf{i}^T)}{q^{N-1}} \\ &= \sum_{\mathbf{i}\in\mathbb{F}_q} \log p(\mathbf{i}\mathbf{D}) \log r(\mathbf{a}\mathbf{i}^T) - \frac{1}{q^N}\sum_{\mathbf{i}\in\mathbb{F}_q} \log p(\mathbf{i}\mathbf{D}) \sum_{\mathbf{i}\in\mathbb{F}_q} \log r(\mathbf{a}\mathbf{i}^T)\end{aligned}$$

Let $\mathbf{j}$ be a vector in $\mathbb{F}_q^N$. For each $\mathbf{j}$ vector there are $q^{N-\text{rank}(\mathbf{D})}$ $\mathbf{i}$ vectors in $\mathbb{F}_q^N$ satisfying the relation

$$\mathbf{jD} = \mathbf{iD}, \tag{A.3}$$

where rank $(\mathbf{D})$ denotes the rank of the dependency matrix $\mathbf{D}$. Therefore, the first summation in (A.3) is equal to the following nested summation.

$$\begin{aligned}\sum_{\mathbf{i}\in\mathbb{F}_q} \log p(\mathbf{iD}) \log r(\mathbf{ai}^T) &= \frac{1}{q^{N-\text{rank}(\mathbf{D})}} \sum_{\mathbf{i}\in\mathbb{F}_q^N} \left(\sum_{\mathbf{j}\in\mathbb{F}_q^N : \mathbf{iD}=\mathbf{jD}} \log p(\mathbf{jD}) \log r(\mathbf{aj}^T) \right) \\ &= \frac{1}{q^{N-\text{rank}(\mathbf{D})}} \sum_{\mathbf{i}\in\mathbb{F}_q^N} \log p(\mathbf{iD}) \left(\sum_{\mathbf{j}\in\mathbb{F}_q^N : \mathbf{iD}=\mathbf{jD}} \log r(\mathbf{aj}^T) \right)\end{aligned}$$

The inner summation on the right hand side above can be grouped as

$$\begin{aligned}\sum_{\mathbf{i}\in\mathbb{F}_q} \log p(\mathbf{iD}) \log r(\mathbf{ai}^T) &= \frac{1}{q^{N-\text{rank}(\mathbf{D})}} \sum_{\mathbf{i}\in\mathbb{F}_q^N} \log p(\mathbf{iD}) \left(\sum_{k\in\mathbb{F}_q} \left(\sum_{\mathbf{j}\in\mathbb{F}_q^N : \mathbf{iD}=\mathbf{jD} \wedge \mathbf{aj}^T=k} \log r(\mathbf{aj}^T) \right) \right) \\ &= \frac{1}{q^{N-\text{rank}(\mathbf{D})}} \sum_{\mathbf{i}\in\mathbb{F}_q^N} \log p(\mathbf{iD}) \left(\sum_{k\in\mathbb{F}_q} \log r(k) \left(\sum_{\mathbf{j}\in\mathbb{F}_q^N : \mathbf{iD}=\mathbf{jD} \wedge \mathbf{aj}^T=k} 1 \right) \right).\end{aligned}$$

We have to determine how many times the innermost summation above runs. Let $\mathbf{d}_1$, $\mathbf{d}_2$, …, $\mathbf{d}_{\text{rank}(\mathbf{D})}$ be the nonzero rows of $\mathbf{D}$. Then the innermost summation above runs once for all $\mathbf{j}$ vector satisfying the system of linear equations below.

$$\begin{bmatrix} \mathbf{d}_1 \\ \mathbf{d}_2 \\ \vdots \\ \mathbf{d}_{\text{rank}(\mathbf{D})} \\ \mathbf{a} \end{bmatrix} \mathbf{j}^T = \begin{bmatrix} \mathbf{d}_1\mathbf{i}^T \\ \mathbf{d}_2\mathbf{i}^T \\ \vdots \\ \mathbf{d}_{\text{rank}(\mathbf{D})}\mathbf{i}^T \\ i \end{bmatrix}$$

Due to the definition of the dependency matrix, all nonzero rows of $\mathbf{D}$ are linearly independent. Moreover, all these nonzero rows of $\mathbf{D}$ are also linearly independent with $\mathbf{a}$, since $\mathbf{a}$ is not equal to $\mathbf{aD}$. Therefore, the system of linear equations above has $q^{N-\text{rank}(D)-1}$ solutions. Hence,

$$\sum_{\mathbf{i}\in\mathbb{F}_q} \log p(\mathbf{iD}) \log r(\mathbf{ai}^T) = \frac{1}{q} \sum_{\mathbf{i}\in\mathbb{F}_q^N} \log p(\mathbf{iD}) \sum_{k\in\mathbb{F}_q} \log r(k).$$

Inserting this result into (A.3) yields

$$\begin{aligned}< p(\mathbf{x}), C_{\mathbb{F}_q^N}\left\{r(\mathbf{ax}^T)\right\} > &= \frac{1}{q} \sum_{\mathbf{i}\in\mathbb{F}_q^N} \log p(\mathbf{iD}) \sum_{k\in\mathbb{F}_q} \log r(k) - \frac{1}{q^N} \sum_{\mathbf{i}\in\mathbb{F}_q} \log p(\mathbf{iD}) \sum_{\mathbf{i}\in\mathbb{F}_q} \log r(\mathbf{ai}^T) \\ &= \frac{1}{q} \sum_{\mathbf{i}\in\mathbb{F}_q^N} \log p(\mathbf{iD}) \sum_{k\in\mathbb{F}_q} \log r(k) - \frac{1}{q} \sum_{\mathbf{i}\in\mathbb{F}_q^N} \log p(\mathbf{iD}) \sum_{k\in\mathbb{F}_q} \log r(k) \\ &= 0,\end{aligned}$$

which completes the proof.

A.4 Proofs and Derivations in Chapter 6

A.4.1 The factorization of a posteriori probability of X given in Section 6.2

Expanding the absolute value in (6.3) yields

$$\begin{aligned} p(\mathbf{x}) &= C_{\mathbb{F}_q^N}\left\{\exp\left(-\frac{1}{2\sigma^2}\left|y-\sum_{i=1}^{N}h_i\mu_q\left(x_i\right)\right|^2\right)\right\} \\ &= C_{\mathbb{F}_q^N}\left\{\exp\left(-\frac{1}{2\sigma^2}\left(|y|^2-2y\sum_{i=1}^{N}\mathrm{Re}\left\{h_i^*\mu_q\left(x_i\right)^*\right\}+\left|\sum_{i=1}^{N}h_i\mu_q\left(x_i\right)\right|^2\right)\right)\right\}. \end{aligned}$$

Since $|y|^2$ does not depend on $\mathbf{x}$, it can be cancelled by the normalization operator which gives,

$$\begin{aligned} p(\mathbf{x}) &= C_{\mathbb{F}_q^N}\left\{\exp\left(\frac{1}{2\sigma^2}\left(\sum_{i=1}^{N}2\mathrm{Re}\left\{yh_i^*\mu_q\left(x_i\right)^*\right\}-\left(\sum_{i=1}^{N}h_i\mu_q\left(x_i\right)\right)\left(\sum_{i=1}^{N}h_i^*\mu_q\left(x_i\right)^*\right)\right)\right)\right\} \\ &= C_{\mathbb{F}_q^N}\left\{\exp\left(\sum_{i=1}^{N}\frac{2\mathrm{Re}\left\{yh_i^*\mu_q\left(x_i\right)^*\right\}-\left|h_i\mu_q\left(x_i\right)\right|^2}{2\sigma^2}-\sum_{j=2}^{N}\sum_{i=1}^{j-1}\frac{2\mathrm{Re}\left\{h_i\mu_q\left(x_i\right)h_j^*\mu_q\left(x_j\right)^*\right\}}{2\sigma^2}\right)\right\} \end{aligned}$$

Since PSK is a constant amplitude modulation, $\left|\mu_q\left(x_i\right)\right|$ is constant for all x_i. Consequently, $\left|h_i\mu_q\left(x_i\right)\right|^2$ does not depend on $\mathbf{x}$. Canceling $\left|h_i\mu_q\left(x_i\right)\right|^2$ by the normalization operator yields the desired factorization.

$$p(\mathbf{x}) = C_{\mathbb{F}_q^N}\left\{\prod_{i=1}^{N}\exp\left(\frac{2\mathrm{Re}\left\{yh_i^*\mu_q\left(x_i\right)^*\right\}}{2\sigma^2}\right)\prod_{j=2}^{N}\prod_{i=1}^{j-1}\exp\left(-\frac{2\mathrm{Re}\left\{h_ih_j^*\mu_q\left(x_i\right)\mu_q\left(x_j\right)^*\right\}}{2\sigma^2}\right)\right\} \quad \text{(A.4)}$$

A.4.2 Proof of Theorem 6.1

The necessary row operations are listed below.

1. Add 1^{st} row to $(1+\frac{(j-2)(j-1)}{2})^{th}$ row for $j=3$ up to N.
2. For $i=4$ up to N, add $(\frac{(i-1)(i-2)}{2}+3)^{th}$ row to
   (a) $(\frac{(i-1)(i-2)}{2}+1)^{th}$ row,
   (b) $(\frac{(i-1)(i-2)}{2}+2)^{th}$ row,
   (c) $(\frac{(i-1)(i-2)}{2}+j)^{th}$ row for $j=4$ up to $i-1$,

(d) $(\frac{(i)(i-1)}{2}+3)^{th}$ row,

(e) $(\frac{(j-1)(j-2)}{2}+1+i)^{th}$ row for $j = i+2$ up to N.

A.4.3 Derivation of the factorization in (6.51)

$$\begin{aligned}\Pr\{\mathbf{X}=\mathbf{x}|\mathbf{Y}=\mathbf{y}\} &= C_{\mathbb{F}_q^{2N_t}}\left\{\exp\left(-\frac{\|\mathbf{y}-\mathbf{H}_c\mathbf{w}\|^2}{2\sigma^2}\right)\right\}\\ &\propto \exp\left(-\frac{1}{2\sigma^2}\left(\|\mathbf{y}\|^2-2\mathrm{Re}\left\{\mathbf{w}^H\mathbf{H}_c^H\mathbf{y}\right\}+\mathbf{w}^H\mathbf{H}_c^H\mathbf{H}_c\mathbf{w}\right)\right)\end{aligned}$$

We can cancel $\|y\|^2$ since it is constant for all $\mathbf{x}$. Let $\mathbf{u} \triangleq \mathbf{H}_c^H\mathbf{y}$ and $\mathbf{R} \triangleq \mathbf{H}_c^H\mathbf{H}_c$. Then the factorization becomes,

$$\begin{aligned}\Pr\{\mathbf{X}=\mathbf{x}|\mathbf{Y}=\mathbf{y}\} &\propto \exp\left(-\frac{1}{2\sigma^2}\left(-2\mathrm{Re}\left\{\mathbf{w}^H\mathbf{u}\right\}+\mathbf{w}^H\mathbf{R}\mathbf{w}\right)\right)\\ &\propto \exp\left(\frac{1}{2\sigma^2}\left(2\sum_{k=1}^{N_t}\mathrm{Re}\left\{\nu(\mathbf{x}_k)\,u_k^*\right\}-\sum_{k=1}^{N_t}\sum_{l=1}^{N_t}\nu(\mathbf{x}_k)^*(\mathbf{R})_{k,l}\nu(\mathbf{x}_l)\right)\right),\end{aligned}$$

where u_k is the k^{th} component of $\mathbf{u}$ and $(\mathbf{R})_{k,l}$ is the entry in the k^{th} row and j^{th} column of the matrix $\mathbf{R}$. Since $\mathbf{R}$ is hermitian symmetric,

$$\begin{aligned}\Pr\{\mathbf{X}=\mathbf{x}|\mathbf{Y}=\mathbf{y}\} &\propto \exp\left(\frac{1}{2\sigma^2}\left(\sum_{k=1}^{N_t}\left(2\mathrm{Re}\left\{\nu(\mathbf{x}_k)\,u_k^*\right\}-\|\nu(\mathbf{x}_k)\|^2(\mathbf{R})_{k,k}\right)\right)\right)\\ &\quad\cdot\exp\left(-\frac{1}{2\sigma^2}\left(\sum_{k=2}^{N_t}\sum_{l=1}^{k-1}2\mathrm{Re}\left\{\nu(\mathbf{x}_k)^*(\mathbf{R})_{k,l}\nu(\mathbf{x}_l)\right\}\right)\right).\end{aligned}$$

Since $\|\nu(\mathbf{x}_k)\|^2$ is constant,

$$\Pr\{\mathbf{X}=\mathbf{x}|\mathbf{Y}=\mathbf{y}\} \propto \exp\left(\frac{1}{2\sigma^2}\left(\sum_{k=1}^{N_t}2\mathrm{Re}\left\{\nu(\mathbf{x}_k)\,u_k^*\right\}-\sum_{k=2}^{N_t}\sum_{l=1}^{k-1}2\mathrm{Re}\left\{\nu(\mathbf{x}_k)^*(\mathbf{R})_{k,l}\nu(\mathbf{x}_l)\right\}\right)\right). \quad (A.5)$$

The function $\nu(\mathbf{x}_k)$ can be expressed in terms of $\beta(.)$ function as

$$\nu(\mathbf{x}_k) = a\beta(x_{2k-1}) + a^*\beta(x_{2k}),$$

where $a = \frac{1}{2} + j\frac{1}{2}$. Therefore,

$$\mathrm{Re}\left\{\nu(\mathbf{x}_k)\,u_k^*\right\} = \mathrm{Re}\left\{au_k^*\right\}\beta(x_{2k-1}) + \mathrm{Re}\left\{a^*u_k^*\right\}\beta(x_{2k}). \quad (A.6)$$

Furthermore,

$$\begin{aligned}
\mathrm{Re}\left\{\nu\left(\mathbf{x}_k\right)^*\left(\mathbf{R}\right)_{k,l}\nu\left(\mathbf{x}_l\right)\right\} &= \mathrm{Re}\left\{\left(a^*\beta\left(x_{2k-1}\right)+a\beta\left(x_{2k}\right)\right)\left(\mathbf{R}\right)_{k,l}\left(a\beta\left(x_{2l-1}\right)+a^*\beta\left(x_{2l}\right)\right)\right\} \\
&= \mathrm{Re}\left\{\left|a\right|^2\left(\mathbf{R}\right)_{k,l}\beta\left(x_{2k-1}\right)\beta\left(x_{2l-1}\right)\right\}+\mathrm{Re}\left\{\left(a^*\right)^2\left(\mathbf{R}\right)_{k,l}\beta\left(x_{2k-1}\right)\beta\left(x_{2l}\right)\right\} \\
&\quad +\mathrm{Re}\left\{a^2\left(\mathbf{R}\right)_{k,l}\beta\left(x_{2k}\right)\beta\left(x_{2l-1}\right)\right\}+\mathrm{Re}\left\{\left|a\right|^2\left(\mathbf{R}\right)_{k,l}\beta\left(x_{2k}\right)\beta\left(x_{2l}\right)\right\} \\
&= \frac{1}{2}(\beta\left(x_{2k-1}+x_{2l-1}\right)\mathrm{Re}\left\{\left(\mathbf{R}\right)_{k,l}\right\}+\beta\left(x_{2k-1}+x_{2l}\right)\mathrm{Im}\left\{\left(\mathbf{R}\right)_{k,l}\right\} \\
&\quad -\beta\left(x_{2k}+x_{2l-1}\right)\mathrm{Im}\left\{\left(\mathbf{R}\right)_{k,l}\right\}+\beta\left(x_{2k}+x_{2l}\right)\mathrm{Re}\left\{\left(\mathbf{R}\right)_{k,l}\right\}). \qquad \text{(A.7)}
\end{aligned}$$

Inserting (A.6) and (A.7) together with the definition of the $\gamma(.;.)$ function into (A.5) gives the desired factorization.

$$\begin{aligned}
\Pr\{\mathbf{X}=\mathbf{x}|\mathbf{Y}=\mathbf{y}\} \propto & \prod_{k=1}^{N_t}\gamma\left(x_{2k-1};\frac{\mathrm{Re}\left\{u_k\right\}+\mathrm{Im}\left\{u_k\right\}}{2},\sigma\right)\gamma\left(x_{2k};\frac{\mathrm{Re}\left\{u_k\right\}-\mathrm{Im}\left\{u_k\right\}}{2},\sigma\right) \\
& \cdot\prod_{k=2}^{N_t}\prod_{l=1}^{k-1}\gamma\left(x_{2k-1}+x_{2l-1};-\frac{\mathrm{Re}\left\{\left(\mathbf{R}\right)_{k,l}\right\}}{2},\sigma\right)\gamma\left(x_{2k-1}+x_{2l};-\frac{\mathrm{Im}\left\{\left(\mathbf{R}\right)_{k,l}\right\}}{2},\sigma\right). \\
& \cdot\prod_{k=2}^{N_t}\prod_{l=1}^{k-1}\gamma\left(x_{2k}+x_{2l-1};\frac{\mathrm{Im}\left\{\left(\mathbf{R}\right)_{k,l}\right\}}{2},\sigma\right)\gamma\left(x_{2k}+x_{2l};-\frac{\mathrm{Re}\left\{\left(\mathbf{R}\right)_{k,l}\right\}}{2},\sigma\right)
\end{aligned} \qquad \text{(A.8)}$$

A.4.4 Permutations used in the simulation in Section 6.6.5

The set $\mathcal{P}$ consists of the following permutations.

$$\begin{aligned}
\mathbf{P}_1 &= \left[\mathbf{f}_1^T\ \mathbf{f}_3^T\ \mathbf{f}_5^T\ \mathbf{f}_7^T\ \mathbf{f}_9^T\ \mathbf{f}_{11}^T\ \mathbf{f}_{13}^T\ \mathbf{f}_{15}^T\ \mathbf{f}_2^T\ \mathbf{f}_4^T\ \mathbf{f}_6^T\ \mathbf{f}_8^T\ \mathbf{f}_{10}^T\ \mathbf{f}_{12}^T\ \mathbf{f}_{14}^T\ \mathbf{f}_{16}^T\right] \\
\mathbf{P}_2 &= \left[\mathbf{f}_1^T\ \mathbf{f}_3^T\ \mathbf{f}_5^T\ \mathbf{f}_7^T\ \mathbf{f}_9^T\ \mathbf{f}_{11}^T\ \mathbf{f}_{13}^T\ \mathbf{f}_{15}^T\ \mathbf{f}_4^T\ \mathbf{f}_6^T\ \mathbf{f}_8^T\ \mathbf{f}_{10}^T\ \mathbf{f}_{12}^T\ \mathbf{f}_{14}^T\ \mathbf{f}_{16}^T\ \mathbf{f}_2^T\right] \\
\mathbf{P}_3 &= \left[\mathbf{f}_1^T\ \mathbf{f}_3^T\ \mathbf{f}_5^T\ \mathbf{f}_7^T\ \mathbf{f}_9^T\ \mathbf{f}_{11}^T\ \mathbf{f}_{13}^T\ \mathbf{f}_{15}^T\ \mathbf{f}_6^T\ \mathbf{f}_8^T\ \mathbf{f}_{10}^T\ \mathbf{f}_{12}^T\ \mathbf{f}_{14}^T\ \mathbf{f}_{16}^T\ \mathbf{f}_2^T\ \mathbf{f}_4^T\ \right] \\
\mathbf{P}_4 &= \left[\mathbf{f}_1^T\ \mathbf{f}_3^T\ \mathbf{f}_5^T\ \mathbf{f}_7^T\ \mathbf{f}_9^T\ \mathbf{f}_{11}^T\ \mathbf{f}_{13}^T\ \mathbf{f}_{15}^T\ \mathbf{f}_8^T\ \mathbf{f}_{10}^T\ \mathbf{f}_{12}^T\ \mathbf{f}_{14}^T\ \mathbf{f}_{16}^T\ \mathbf{f}_2^T\ \mathbf{f}_4^T\ \mathbf{f}_6^T\ \right]
\end{aligned}$$

$$\begin{aligned}
\mathbf{P}_5 &= \left[\mathbf{f}_1^T\ \mathbf{f}_3^T\ \mathbf{f}_5^T\ \mathbf{f}_7^T\ \mathbf{f}_9^T\ \mathbf{f}_{11}^T\ \mathbf{f}_{13}^T\ \mathbf{f}_{15}^T\ \mathbf{f}_{10}^T\ \mathbf{f}_{12}^T\ \mathbf{f}_{14}^T\ \mathbf{f}_{16}^T\ \mathbf{f}_2^T\ \mathbf{f}_4^T\ \mathbf{f}_6^T\ \mathbf{f}_8^T\right] \\
\mathbf{P}_6 &= \left[\mathbf{f}_1^T\ \mathbf{f}_3^T\ \mathbf{f}_5^T\ \mathbf{f}_7^T\ \mathbf{f}_9^T\ \mathbf{f}_{11}^T\ \mathbf{f}_{13}^T\ \mathbf{f}_{15}^T\ \mathbf{f}_{12}^T\ \mathbf{f}_{14}^T\ \mathbf{f}_{16}^T\ \mathbf{f}_2^T\ \mathbf{f}_4^T\ \mathbf{f}_6^T\ \mathbf{f}_8^T\ \mathbf{f}_{10}^T\right] \\
\mathbf{P}_7 &= \left[\mathbf{f}_1^T\ \mathbf{f}_3^T\ \mathbf{f}_5^T\ \mathbf{f}_7^T\ \mathbf{f}_9^T\ \mathbf{f}_{11}^T\ \mathbf{f}_{13}^T\ \mathbf{f}_{15}^T\ \mathbf{f}_{14}^T\ \mathbf{f}_{16}^T\ \mathbf{f}_2^T\ \mathbf{f}_4^T\ \mathbf{f}_6^T\ \mathbf{f}_8^T\ \mathbf{f}_{10}^T\ \mathbf{f}_{12}^T\right] \\
\mathbf{P}_8 &= \left[\mathbf{f}_1^T\ \mathbf{f}_3^T\ \mathbf{f}_5^T\ \mathbf{f}_7^T\ \mathbf{f}_9^T\ \mathbf{f}_{11}^T\ \mathbf{f}_{13}^T\ \mathbf{f}_{15}^T\ \mathbf{f}_{16}^T\ \mathbf{f}_2^T\ \mathbf{f}_4^T\ \mathbf{f}_6^T\ \mathbf{f}_8^T\ \mathbf{f}_{10}^T\ \mathbf{f}_{12}^T\ \mathbf{f}_{14}^T\right]
\end{aligned}$$

$$
\begin{aligned}
\mathbf{P}_9 &= \left[\mathbf{f}_4^T\ \mathbf{f}_6^T\ \mathbf{f}_8^T\ \mathbf{f}_{10}^T\ \mathbf{f}_{12}^T\ \mathbf{f}_{14}^T\ \mathbf{f}_{16}^T\ \mathbf{f}_2^T\ \mathbf{f}_1^T\ \mathbf{f}_3^T\ \mathbf{f}_5^T\ \mathbf{f}_7^T\ \mathbf{f}_9^T\ \mathbf{f}_{11}^T\ \mathbf{f}_{13}^T\ \mathbf{f}_{15}^T\right] \\
\mathbf{P}_{10} &= \left[\mathbf{f}_6^T\ \mathbf{f}_8^T\ \mathbf{f}_{10}^T\ \mathbf{f}_{12}^T\ \mathbf{f}_{14}^T\ \mathbf{f}_{16}^T\ \mathbf{f}_2^T\ \mathbf{f}_4^T\ \mathbf{f}_1^T\ \mathbf{f}_3^T\ \mathbf{f}_5^T\ \mathbf{f}_7^T\ \mathbf{f}_9^T\ \mathbf{f}_{11}^T\ \mathbf{f}_{13}^T\ \mathbf{f}_{15}^T\right] \\
\mathbf{P}_{11} &= \left[\mathbf{f}_8^T\ \mathbf{f}_{10}^T\ \mathbf{f}_{12}^T\ \mathbf{f}_{14}^T\ \mathbf{f}_{16}^T\ \mathbf{f}_2^T\ \mathbf{f}_4^T\ \mathbf{f}_6^T\ \mathbf{f}_1^T\ \mathbf{f}_3^T\ \mathbf{f}_5^T\ \mathbf{f}_7^T\ \mathbf{f}_9^T\ \mathbf{f}_{11}^T\ \mathbf{f}_{13}^T\ \mathbf{f}_{15}^T\right] \\
\mathbf{P}_{12} &= \left[\mathbf{f}_{10}^T\ \mathbf{f}_{12}^T\ \mathbf{f}_{14}^T\ \mathbf{f}_{16}^T\ \mathbf{f}_2^T\ \mathbf{f}_4^T\ \mathbf{f}_6^T\ \mathbf{f}_8^T\ \mathbf{f}_1^T\ \mathbf{f}_3^T\ \mathbf{f}_5^T\ \mathbf{f}_7^T\ \mathbf{f}_9^T\ \mathbf{f}_{11}^T\ \mathbf{f}_{13}^T\ \mathbf{f}_{15}^T\right] \\
\\
\mathbf{P}_{13} &= \left[\mathbf{f}_{12}^T\ \mathbf{f}_{14}^T\ \mathbf{f}_{16}^T\ \mathbf{f}_2^T\ \mathbf{f}_4^T\ \mathbf{f}_6^T\ \mathbf{f}_8^T\ \mathbf{f}_{10}^T\ \mathbf{f}_1^T\ \mathbf{f}_3^T\ \mathbf{f}_5^T\ \mathbf{f}_7^T\ \mathbf{f}_9^T\ \mathbf{f}_{11}^T\ \mathbf{f}_{13}^T\ \mathbf{f}_{15}^T\right] \\
\mathbf{P}_{14} &= \left[\mathbf{f}_{14}^T\ \mathbf{f}_{16}^T\ \mathbf{f}_2^T\ \mathbf{f}_4^T\ \mathbf{f}_6^T\ \mathbf{f}_8^T\ \mathbf{f}_{10}^T\ \mathbf{f}_{12}^T\ \mathbf{f}_1^T\ \mathbf{f}_3^T\ \mathbf{f}_5^T\ \mathbf{f}_7^T\ \mathbf{f}_9^T\ \mathbf{f}_{11}^T\ \mathbf{f}_{13}^T\ \mathbf{f}_{15}^T\right] \\
\mathbf{P}_{15} &= \left[\mathbf{f}_{16}^T\ \mathbf{f}_2^T\ \mathbf{f}_4^T\ \mathbf{f}_6^T\ \mathbf{f}_8^T\ \mathbf{f}_{10}^T\ \mathbf{f}_{12}^T\ \mathbf{f}_{14}^T\ \mathbf{f}_1^T\ \mathbf{f}_3^T\ \mathbf{f}_5^T\ \mathbf{f}_7^T\ \mathbf{f}_9^T\ \mathbf{f}_{11}^T\ \mathbf{f}_{13}^T\ \mathbf{f}_{15}^T\right]
\end{aligned}
$$

A.5 Proofs and Derivations in Chapter 7

A.5.1 Proof of Theorem 7.1

The proof in the forward direction is actually an implication of the cut-set independence theorem stated in [2]. An alternative proof is given below.

Let $t_k(\mathbf{x})$ and $t_l(\mathbf{x})$ be defined as

$$t_k(\mathbf{x}) \triangleq C_{\mathbb{F}_q^N}\left\{\prod_{\mathbf{a}_i\in\mathcal{K}_k} r_i(\mathbf{a}_i\mathbf{x}^T)\right\}, \tag{A.9}$$

$$t_l(\mathbf{x}) \triangleq C_{\mathbb{F}_q^N}\left\{\prod_{\mathbf{a}_i\in\mathcal{K}_l\setminus\mathcal{K}_k} r_i(\mathbf{a}_i\mathbf{x}^T)\right\}. \tag{A.10}$$

Clearly,

$$p(\mathbf{x}) = C_{\mathbb{F}_q^N}\left\{t_k(\mathbf{x})t_l(\mathbf{x})\right\}. \tag{A.11}$$

Due to the definitions of $\mathcal{K}_k$ and $\mathcal{K}_k$, $t_k(\mathbf{x})$ and $t_l(\mathbf{x})$ satisfies

$$r_k(\mathbf{x}) \triangleq r_k(\mathbf{x}(\mathbf{I}-\mathbf{E}_k)), \tag{A.12}$$

$$r_l(\mathbf{x}) \triangleq r_l(\mathbf{x}(\mathbf{I}-\mathbf{E}_l)), \tag{A.13}$$

where $\mathbf{E}_k$ ($\mathbf{E}_l$) is the dependency with just a single 1 on its k^{th} (l^{th}) entry on the main diagonal.

An equivalent requirement on conditional independence can be obtained by multiplying both sides of (7.1) with $\Pr\{\mathbf{X}_{\setminus\{k,l\}} = \mathbf{x}_{\setminus\{k,l\}}\}\Pr\{\mathbf{X}_{\setminus\{k\}} = \mathbf{x}_{\setminus\{l\}}\}$ as follows.

$$p(\mathbf{x})\Pr\{\mathbf{X}_{\setminus\{k,l\}} = \mathbf{x}_{\setminus\{k,l\}}\} = \Pr\{\mathbf{X}_{\setminus\{k\}} = \mathbf{x}_{\setminus\{k\}}\}\Pr\{\mathbf{X}_{\setminus\{l\}} = \mathbf{x}_{\setminus\{l\}}\} \tag{A.14}$$

The marginal distribution $\Pr\{\mathbf{X}_{\setminus\{k\}} = \mathbf{x}_{\setminus\{k\}}\}$ can be derived in terms of $t_k(\mathbf{x})$ and $t_l(\mathbf{x})$ as

$$\begin{aligned}\Pr\{\mathbf{X}_{\setminus\{k\}} = \mathbf{x}_{\setminus\{k\}}\} &= \sum_{\forall x_k \in \mathbb{F}_q} p(\mathbf{x}) \\ &= C_{\mathbb{F}_q^{n-1}} \left\{ \sum_{\forall x_k \in \mathbb{F}_q} r_k(\mathbf{x}) r_l(\mathbf{x}) \right\} \\ &= C_{\mathbb{F}_q^{n-1}} \left\{ r_k(\mathbf{x}) \sum_{\forall x_k \in \mathbb{F}_q} r_l(\mathbf{x}) \right\}, \end{aligned} \tag{A.15}$$

where the last line follows from (A.12). Other marginal distributions in (A.14) can similarly be derived as

$$\Pr\{\mathbf{X}_{\setminus\{l\}} = \mathbf{x}_{\setminus\{l\}}\} = C_{\mathbb{F}_q^{n-1}} \left\{ r_l(\mathbf{x}) \sum_{\forall x_l \in \mathbb{F}_q} r_k(\mathbf{x}) \right\} \tag{A.16}$$

$$\Pr\{\mathbf{X}_{\setminus\{k,l\}} = \mathbf{x}_{\setminus\{k,l\}}\} = C_{\mathbb{F}_q^{n-2}} \left\{ \sum_{\forall x_k \in \mathbb{F}_q} r_l(\mathbf{x}) \sum_{\forall x_l \in \mathbb{F}_q} r_k(\mathbf{x}) \right\} \tag{A.17}$$

Inserting (A.11), (A.15), (A.16), and (A.17) into (A.14) verifies that the equality in (A.14) holds and completes the proof in the forward direction.

The proof in the backward direction starts with multiplying both sides of (7.1) with $\Pr\{\mathbf{X}_{\setminus\{k\}} = \mathbf{x}_{\setminus\{k\}}\}$ which yields

$$\begin{aligned} p(\mathbf{x}) &= \Pr\{\mathbf{X}_{\setminus\{k\}} = \mathbf{x}_{\setminus\{k\}}\} \Pr\{X_k = x_k | \mathbf{X}_{\setminus\{k,l\}} = \mathbf{k}, \mathbf{l}_{\setminus\{\}}\} \\ &= C_{\mathbb{F}_q^N} \{m_k(\mathbf{x}) m_l(\mathbf{x})\}, \end{aligned} \tag{A.18}$$

where $m_k(\mathbf{x})$ and $m_l(\mathbf{x})$ are $C_{\mathbb{F}_q^N} \{\Pr\{\mathbf{X}_{\setminus\{k\}} = \mathbf{x}_{\setminus\{k\}}\}\}$ and $\Pr\{X_k = x_k | \mathbf{X}_{\setminus\{k,l\}} = \mathbf{k}, \mathbf{l}_{\setminus\{\}}\}$. Clearly, these functions satisfy

$$m_k(\mathbf{x}) = m_k(\mathbf{x}(\mathbf{I} - \mathbf{E}_k)), \tag{A.19}$$

$$m_l(\mathbf{x}) = m_l(\mathbf{x}(\mathbf{I} - \mathbf{E}_l)). \tag{A.20}$$

Then due to Theorem 4.4 $p(\mathbf{x})$ can be factored as in (7.2).

VITA

Personal Information

Date and Place of Birth:	June 12, 1980 — İstanbul, Turkey
Nationality:	Turkish
Marital Status:	Married
Address:	Elektrik Elektronik Müh. Bölümü, ODTÜ, 06531, Ankara/Turkey
Phone:	+90 505 514 61 14
Web Page:	www.eee.metu.edu.tr/~fatih
e-mail:	fatih@eee.metu.edu.tr
	bfatih@gmail.com

Education

Sep. 2005 *M.Sc. in Electrical and Electronics Engineering*
Middle East Technical University, Ankara, Turkey
Thesis Title: *"Sub-Graph Approach in Iterative Sum-Product Algorithm"*
Advisor: Prof. Dr. Buyurman Baykal
Co-advisor: Assoc. Prof. Dr. Ali Özgür Yılmaz

June 2002 *B.Sc. in Electrical and Electronics Engineering*
Middle East Technical University, Ankara, Turkey

Teaching Experience

Sep. 2002 - Sep. 2009	*Teaching Assistant* Department of Electrical and Electronics Engineering Middle East Technical University Signal processing courses Non-linear electronics for communications course Analog electronics course Electrical circuits laboratory course
Dec. 2007	*Instructor of a short course on signal processing* Environmental Tectonics Corporation, Turkey Branch
Oct. 2005	*Establishment of the electrical circuits laboratory* Northern Cyprus Campus of Middle East Technical University

Engineering Experience

Oct. 2001- May 2002	*DSP Programmer* Implementation of NATO STANAG 4285 (HF modem standard) on TMS320C54 DSP processor
Sep. 2007- March 2009	*Researcher* Detecting and classifying low probability of intercept radar project

Publications

- Bayramoğlu M. F. and Yılmaz A. Ö., *"An analysis method of multivariate probability mass functions"* , to be submitted
- Bayramoğlu M. F. and Yılmaz A. Ö., *"Klasik tespit kuramı için Öklid geometrisi ile genel bir gösterim"*, IEEE 18. Sinyal İşleme ve İletişim Uygulamaları Kurultayı (SİU), April 2010, Diyarbakır, Turkey, An English version is available on arXiv.

- Bayramoğlu M. F. and Yılmaz A. Ö., *"Factorization of joint probability mass functions into parity check interactions"*, Int. Symposium on Information Theory, June 2009, Seoul, Korea

- Bayramoğlu M. F. and Yılmaz A. Ö., *"A Hilbert space of probability mass functions and applications on the sum-product algorithm"*, Int. Symposium on Turbo Codes, September 2008, Lausanne, Switzerland

- Bayramoğlu M. F., Yılmaz A. Ö., and Baykal B., *"Sub-graph approach in iterative sum-product algorithm"*, Int. Symposium on Turbo Codes, April 2006, Munich, Germany

Honors and Awards

1998	Ranked 27^{th} in Nationwide university admission examination among 1.3 million exam takers
1997	Silver medalist in National Olympics in Informatics

Computer Skills

C++, Matlab, Linux, Agilent VEE, Texas Instruments TMS320C54 Assembly language, LaTeX

Interests and Activities

Road cycling, archery, history