The Dissertation Committee for Anindya Chandra Patthak
certifies that this is the approved version of the following dissertation:

Error Correcting Codes : Local Testing, List Decoding, and Applications

Committee:

David I. Zuckerman, Supervisor

Anna Gál

Charanjit S. Jutla

Adam Klivans

José Felipe Voloch

Error Correcting Codes : Local Testing, List Decoding, and Applications

by

Anindya Chandra Patthak, B. Tech.

Dissertation

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

Doctor of Philosophy

The University of Texas at Austin

December 2007

To *Sejomama*

and

the loving memory of *Dadu*

Acknowledgments

First and foremost, this thesis would not have been possible without the support and help of my advisor, David Zuckerman. He has been wonderful over the years, listening to me patiently, giving me advice whenever needed even exceeding the limits of academic boundary, allowing me the flexibility of choosing my own problems, encouraging me when I did not make enough progress, sharing his insights to keep me excited. I would always cherish the memory of spending the most valuable years of my life interacting with him.

I can never overstate the contribution of Charanjit S. Jutla into this thesis. Not only has he collaborated on a vast portion of this thesis, but also his never-ending enthusiasm has kept me motivated on my journeys through my other works. He has always listened to me patiently, has shared with me his deep insights, has made sure that I carry out the things to the dirtiest details.

I am deeply indebted to Felipe Voloch. I have learned a great deal of mathematics interacting with him. On numerous occasion I have inundated him with lists of questions that I had no clue about, and often with ill-framed questions. He has always gone through them patiently, explaining me in details, correcting me whenever necessary, carrying out proofs if needed. My research on list decodable codes would not have been possible without his invaluable advice.

I have benefitted immensely from various courses that I took in the mathematics department. Particularly, John Tate has been a tremendous influence on

me. His simplicity and deep insight have motivated me to learn a few things. It is a sheer pity that I did not learn as much as he had offered. I am also thankful to my other collaborators, Atri Rudra and Venkatesan Guruswami. I would like to thank Anna Gál and Adam Klivans for agreeing to serve in my dissertation committee and also for advice at different stages. I am grateful to Madhu Sudan for allowing me to sit in his class and for sharing his enthusiasm and insights. I would also like to thank Prof. Jayadev Misra for sharing puzzles, often cute proofs and for his advice.

I would like to thank the staff, particularly Gloria and Catherine who have made me meet many a deadlines, at our department. I would also like to thank the support that I have got from my close friends Indrajit Bhattacharya, Anirban Dasgupta, Niloy Jyoti Mitra, Shobhik Mukhopadhay, and Amit Saha.

Nothing would have been possible without the overwhelming support of my family. Debojyoti and Kasturi Lahiri have given me the courage and unconditional support to go ahead. My uncles (particularly Diptish C. Bhaumik) and aunts (particularly Radha Bhaumik) have helped me by making sure that things at home run smoothly. The biggest loss that I ever had was the demise of my (maternal) grand father which took place only a few months back. What I am today is largely due to his foresight, care, love, and affection. Also, the most wonderful thing that has happened in my life took place during these years when I found Shalini, my beloved and best friend. She has been a constant source of motivation. She has been there always through her presence and absence whatever be the need of the hour.

Anindya Chandra Patthak

The University of Texas at Austin
December 2007

Error Correcting Codes : Local Testing, List Decoding, and Applications

Publication No. _____

Anindya Chandra Patthak, Ph.D.
The University of Texas at Austin, 2007

Supervisor: David I. Zuckerman

This dissertation is a study of special types of error correcting codes and their applications. It consists of three parts.

First, we study Generalized Reed-Muller codes (over prime fields), aka low-degree polynomials. Specifically, we show that these codes are locally testable. Locally testable codes are a class of error-correcting codes with the property that given (black-box access to) a word, it is possible to determine with high probability whether the given word is close to a codeword by querying randomly at a sublinear number of places. Such codes are known to be useful for efficient constructions of probabilistically checkable proofs. Our analysis also enables us to obtain a *self-corrector* for the given function, in case the function is reasonably close to a

codeword. Specifically, we show that the value of the function at any given point may be obtained with good probability by querying the function on a few random points. Utilizing pairwise-independence an even higher probability can be achieved by querying the function on slightly more random points and using majority logic decoding. Our result implies that if the acceptance probability is low, then the function is far from low-degree polynomials. Is it possible to estimate the distance even when the received word is far from low-degree polynomials? We could achieve only a conditional result on this front. Specifically, we observe that under certain condition the Gowers uniformity norm estimates the proximity of a function to a low-degree polynomial.

Second, we study efficient constructions of optimal list decodable codes. List decodable codes are error-correcting codes that can deal with highly noisy channels. When a received word has too many errors, unambiguous decoding is no longer possible. A plausible alternative in such a circumstance is to output a small list of possible codewords, each having some minimum agreement with the received word. This is known as the list decoding problem. It is known that good list decodable codes exist. We construct a new family of error-correcting codes based on algebraic curves over finite fields and present efficient list decoding algorithms for the family. These codes extend the class of algebraic-geometric (AG) codes via a generalization of the approach in the recent breakthrough work of Parvaresh and Vardy [PV05].

Third, we develop a new technique to lower-bound the minimum distance of certain types of quasi-cyclic codes with large dimension by reducing the problem to lower-bounding the minimum distance of a few significantly smaller dimensional codes. Using this technique, we prove that a code similar to the SHA-1 (Secure Hash Algorithms) message expansion code has minimum distance at least 82, and that too in just the last 64 of the 80 expanded words. We use this new code to propose an improvement upon the widely used cryptographic hash function SHA-1.

This is particularly important in wake of the recent breakthrough result of Wang et al. [WYY05c, WYY05a] that finds collisions in time much smaller than the naive birthday attack. We expect our technique to be helpful in designing future practical collision-resistant hash functions. We also use this technique to find the minimum weight of the SHA-1 code (25 in last 60 words).

Contents

Chapter 1

Introduction

Coding theory began in the late 1940's with the works of Shannon [Sha48], Golay [Gol49], and Hamming [Ham50]. The original motivation was to correct errors transmitted through noisy communication channels. Since then coding theory has evolved tremendously, benefitting from techniques developed in a wide variety of classical disciplines including combinatorics, probability, algebra, geometry, number theory and theoretical computer science, and has diverse applications in various branches of mathematics, computer science and engineering.

1.0.1 Codes in Theoretical Computer Science

Error-correcting codes have lately become an indispensable tool in complexity theory. In the late 1950's von Neumann [vN56] introduced error-correcting codes to study fault-tolerant computing, where the objective was to design a circuit to compute a given function with high probability even if a fraction of gates in it were faulty. This application was extremely natural considering the fact that codes are designed with redundancies to allow recovery from errors. Later applications exploit specific properties of codes[1]. For example, Shamir's [Sha79] secret sharing scheme uses Reed-Solomon codes, a code well known for its maximum distance separable property. Efficient constructions of k-wise independent (and almost k-wise independent[AGHP92]) probability spaces also use maximum distance separable codes (and codes with large minimum distance, respectively). The study of pseudorandomness and probabilistically checkable proofs use more sophisticated

[1]Throughout this thesis we write code to mean error-correcting code.

codes. The challenge here is not only combinatorial, but also algorithmic. The key properties that have been used in these applications are local testability, local self-correction, local decodability and list decodability. For example, the first two of these properties, i.e., local testability and local correction, are instrumental in getting the remarkable alternate characterization of the complexity class NP (i.e., PCP theorem) and various related hardness results. List decodability and local list decodability have been used extensively in explicit constructions of extractors [Tre01, TZS01, SU01], and in improving worst-case to average-case hardness results.

1.0.2 Overview of This Thesis

This thesis is a study of special types of error-correcting codes and their applications. It consists of three parts. First, we study Generalized Reed-Muller codes (over prime fields) aka low-degree polynomials. Specifically, we [JPRZ04] show that these codes are locally testable. Locally testable codes are a class of error-correcting codes with the property that given (black-box access to) a word, it is possible to determine with high probability whether the given word is close to a codeword by querying randomly at a sublinear number of places. Such codes are known to be useful for efficient constructions of probabilistically checkable proofs. Our analysis also enables us to obtain a *self-corrector* for the given function, in case the function is reasonably close to a codeword. Specifically, we show that the value of the function at any given point may be obtained with good probability by querying the function on a few random points. Utilizing pairwise-independence an even higher probability can be achieved by querying the function on slightly more random points and using majority logic decoding.

Our local testability result implies that if the acceptance probability is low, then the function is far from low-degree polynomials. Is it possible to estimate the distance even when the received word is very far from low-degree polynomials? We could achieve only a conditional result on this front. Specifically, we observe that under certain condition the Gowers uniformity norm (to be defined later) estimates the proximity of a function to low-degree polynomials. These [JPRZ04, JPR04, Pat07] works are mostly done jointly with Charanjit S. Jutla, Atri Rudra and David Zuckerman.

Second, we study efficient constructions of optimal list decodable codes. List decodable codes are error-correcting codes that can deal with highly noisy channels. When a received word has too many errors, unambiguous decoding is no longer possible. A plausible alternative in this circumstance is to output a small list of possible codewords, each having some minimum agreement with the received word. This is known as the list decoding problem. It is known that good list decodable codes exist. We [GP07] define a new family of error-correcting codes based on algebraic curves over finite fields, and develop efficient list decoding algorithms for them. Our codes extend the class of algebraic-geometric codes (henceforth AG codes) via a generalization of the approach in the recent breakthrough work of Parvaresh and Vardy [PV05]. Our work shows that the PV framework applies to fairly general settings by elucidating the key algebraic concepts underlying it. Also, more importantly, AG codes of arbitrary block length exist over *fixed* alphabets, thus enabling us to establish new trade-offs between the list decoding radius and rate over a bounded alphabet size. This [GP07] work is done jointly with Venkatesan Guruswami.

Third, we develop a new technique to lower bound the minimum distance of certain types of quasi-cyclic codes with large dimension by reducing the problem to lower bounding the minimum distance of a few significantly smaller dimensional codes. Using this technique, we prove that a code which is similar to the SHA-1 (Secure Hash Algorithms) message expansion code has minimum distance at least 82, and that too in just the last 64 of the 80 expanded words. Further the minimum weight in the last 60 words (last 48 words) is at least 75 (52 respectively). We use this new code to propose an improvement upon the widely used cryptographic hash function SHA-1. This is particularly important in wake of the recent breakthrough result of Wang et al. [WYY05c, WYY05a] that finds collisions in time much smaller than the naive birthday attack. We expect our technique to be helpful in designing future practical collision-resistant hash functions. We also use this technique to find the minimum weight of the SHA-1 code (25 in last 60 words), which was an open problem. These [JP05b, JP05a, JP05c, JP06] works are done jointly with Charanjit S. Jutla.

Considering the importance and plethora of applications of PCPs and pseudorandomness in complexity theory and cryptography, we believe that a better understanding of locally testable codes and list decodable codes, and codes in general,

will be extremely useful and give further impetus to the field. This thesis is a tiny step in that direction.

1.1 Motivation and Contributions

1.1.1 Local Testability of Generalized Reed-Muller codes

To motivate we begin with a toy problem. The probability of a hard-drive crash is known to be extremely low. Assume we are frequently interested to know whether a given hard-drive is still good. A naive way to do this would be to search the entire hard drive. However, hard drive contains enormous amount of data, and since it crashes with very low probability, reading the whole disk frequently to see whether the data is still valid sounds inefficient.

A more efficient approach would be the following: Assume that we are given a code with the property that probing at random places (probably constantly many or polylogarithmic many) we could determine whether a given word is a valid codeword with high probability. Then we can encode the data in the hard drive using this code. Now we can test efficiently by probing at a few random places to see whether the hard drive is still active. This is the basic idea of local testability. Codes that allow such tests are called **locally testable codes**.

The definition of local testing may sounds optimistic. Is it at all possible to determine whether a given word is a codeword without looking at the whole codeword? How many places do we need to probe? Can the test be done by querying sublinearly or constantly many random places? Of course, there is no a priori reason to believe that any non-trivial code admits such a test. It turns out that there are codes that do admit local tests. The Hadamard code provides a nice example of this class of codes. A codeword in the Hadamard code can be viewed as a linear function. With this view then, a local test would be the following: Pick two uniformly random points x and y and check whether $f(x) + f(y) = f(x + y)$ holds, where $f(x)$ denotes the value of the function at x. Declare f to be a codeword if it holds. First note that it is indeed a local test. Further a thorough analysis [BLR93, BCHS95, BGS98] can be carried out to prove that this test works. In fact, to prove a code locally testable is often quite challenging.

Locally testable codes forms the core in the proof of MIP = NEXPTIME

in [BFL91]. They also play a key role in the development of probabilistically checkable proofs. A probabilistically checkable proof (henceforth PCP) is a robust proof that can be efficiently verified with high probability in a query-efficient manner (a constant number of queries to the proof suffices) by tossing at most logarithmically many coins. This new notion of proof checking yields an alternate characterization of the complexity class NP. The PCP theorem has also played a significant role in establishing various hardness of approximation results ([FGL$^+$91, ALM$^+$92]). Query complexity plays a significant role in obtaining hardness of approximability results for various problems. For example, a query complexity of 3 would translate into a hardness of approximation result for 3-SAT. Also logarithmic many coin tosses implies that the penalty paid for such a super-fast verification is no more than a polynomial factor blow up in the proof size. Due to its connection with hardness results, a lot of attention has been paid to this problem ([BFL91, BFLS91, FGL$^+$91, AS92, RS96, FS95, AKK$^+$03]).

However with the sole exception of [AKK$^+$03], all the above mentioned tests (and their variants) require the degree to be less than the field size. This is because the degree to be tested has to be smaller than the number of points on a line. Hence that approach cannot be used when the degree is larger than the field size.

Alon et al. in [AKK$^+$03] give a tester for the field $\mathbb{F}_2$ without any restriction on the degree. Recall that the collection of polynomials in n variables of degree at most k over $\mathbb{F}_2$ is the Reed-Muller code $\mathrm{RM}(k, n)$ with parameters k and n (see [MS77]). Therefore they essentially prove that Reed-Muller codes are locally testable. Their idea is to pick a random minimum-weight codeword from the dual code and to check if it is orthogonal to the tested vector. It is important to note that these minimum-weight codewords generate the Reed-Muller code.

Working collaboratively with Charanjit S. Jutla, Atri Rudra and David Zuckerman [JPRZ04], we are able to show that Generalized Reed-Muller (henceforth abbreviated as GRM) codes over prime fields are locally-testable. We consider a new basis of GRM code over prime fields that in general differs from the minimum weight basis. This allows us to present a novel exact characterization of multivariate polynomials of degree at most t over prime fields. We are also able to show that the exact characterization can be made robust, (i.e., a probabilistic characterization as needed for the local testability).

The tester of Alon et al. [AKK$^+$03] can also be interpreted as the Gow-

ers norm of certain type. It has been conjectured that these norm defines a good measure, in certain sense, on the distance of a function from low-degree polynomials. Using tools from additive number theory, and the approach of Green and Tao[2][GT05], Samorodnitsky in [Sam07] proves that this is indeed the case for degree one and degree two polynomials. We attempt to generalize his results. However, we could achieve only a conditional result on this front. Specifically, we observe that under certain condition (i.e., availability of certain low end tester for multilinear polynomials) the Gowers uniformity norm estimates the proximity of a function to low-degree polynomials.

1.1.2 List Decoding

Arguably the central question in coding theory is concerned with reliable data recovery. In order to be able to correct the errors, we must have enough information in the uncorrupted region. This implies that redundant information must be added to the data to allow for error recovery. However, with the addition of redundant information, the effective rate of communication decreases. Thus the challenge here is to a design an error-correcting code that allows efficient and reliable data recovery without sacrificing the rate. The design depends on issues like what fraction and what sort of errors can occur.

A received vector is said to have error ϵ fraction if it differs from the closest codeword in at least ϵ fraction of places. It is long known that unique (or unambiguous) decoding is applicable only when the errors are bounded by at most half the distance of the code. In order to correct more errors, it is therefore necessary to allow a small list of possible codewords. This problem is commonly referred to the list decoding problem and has been studied since late 1950's (see [Eli57, Woz58, Eli91]). Codes that allow list decoding are called **list decodable codes**. The study of list decoding has two aspects: combinatorial and algorithmic. The combinatorial aspect essentially deals with what explicit constructions one can hope for, whereas the algorithmic aspects deals with constructions of efficient algorithms to realize the combinatorial bounds.

As hinted previously the error-correction capability also depends on the types of error considered. Shannon considered stochastic channels or equivalently binary

[2]Green and Tao prove the inverse theorem for prime fields with odd characteristics generalizing the celebrated result of Gowers[Gow01].

(and more generally, q-ary) symmetric channel. In this model a q-ary symbol is transmitted without distortion with probability $(1-p)$ and with the remaining probability the symbol is distorted by the channel to a uniform choice among the other $(q-1)$ symbols. In this model Shannon showed that communication is possible with any rate arbitrarily close to $1 - H_q(p)$, the capacity of the underlying communication channel, where

$$H_q(x) \stackrel{\text{def}}{=} x \log_q(q-1) - x \log_q x - (1-x) \log_q(1-x),$$

but not beyond that.

Here we consider the adversarial model. Suppose we encode messages with $\mathcal{R}N$ symbols[3] of information over an alphabet Σ into codewords of N symbols over Σ (here $\mathcal{R}$ is the rate; we think of $\mathcal{R}$ as an absolute constant and let the block length $N \to \infty$). Clearly, to recover the $\mathcal{R}N$ message symbols, we need at least $\mathcal{R}N$ correct symbols at the receiving end. Thus, the absolute information-theoretic limit on fraction of correctable errors is $1 - \mathcal{R}$. We remind the readers that unique decoding can correct only up to half the fraction of this limit. Surprisingly, a notion called list decoding offers the potential to approach this limit (called "capacity"). Under list decoding up to a fraction p of errors, the decoder is required to output a list of all codewords which differ from the received word in at most a fraction p of symbols. The *list size* L needed for the list decoding is the maximum number of codewords that are output in the worst-case. In the limit of $L \to \infty$, there exist list decodable codes of rate $\mathcal{R}$ that can be decoded up to the information-theoretically optimal $1 - \mathcal{R}$ fraction of errors.

The above, however, is a non-constructive result (a combinatorial bound). The codes achieving list decoding capacity were random codes with decoding algorithms no better than exponential-time brute-force search (this is akin to the codes in Shannon's original work for stochastic channels). Though it is known that most codes have this rate vs list-decoding radius trade-off, constructing an explicit code with efficient list-decoding algorithm is not easy. The seminal paper of Sudan [Sud97] shows that Reed-Solomon codes can be efficiently list decoded up to a fraction of $1 - \sqrt{2R}$ errors, where R denotes the rate of the code. Shakrollahi and Wasserman [SW98] generalize Sudan's result to algebraic geometric setting, ob-

[3]We use slightly non-standard symbol as we reserve n and k for different purpose.

taining the same rate versus list-decoding radius trade off with reduced alphabet size. Guruswami and Sudan improve both the results in [GS99]. They show that Reed-Solomon and algebraic Goppa codes both can be efficiently list-decoded up to a fraction of $1 - \sqrt{R}$.

In [CS03] Coppersmith and Sudan give a list-decoding algorithm that corrects a fraction of error up to $1 - \epsilon$ and achieves a rate of $\Omega(\epsilon^{\alpha})$ (for any constant $\alpha > 1$). However, their decoding algorithm is probabilistic and the probability that the correction is achieved is larger than $\Omega(R^{M/(M+1)}))$ assuming a Q-ary symmetric channel.

Recently, building on a line of work in algebraic coding theory [Sud97, GS99, PV05], explicit codes (called folded Reed-Solomon codes) that achieve list decoding capacity with polynomial encoding/decoding complexity are given in by Guruswami and Rudra [GR06].

The work of [GR06] thus meets the challenge of achieving capacity for worst-case errors. However, it has some drawbacks relating to complexity. To correct a fraction $(1 - \mathcal{R} - \epsilon)$ of errors, the proven bound on the worst-case list size of the algorithm in [GR06] is $N^{\Omega(1/\epsilon)}$ where N is the length of the code. In contrast, the existential result gets within ϵ of capacity with list size $O(1/\epsilon)$. It is an important goal to improve the list size to a constant independent of n. The dependence of the list size on n in [GR06] arises because Reed-Solomon codes need an alphabet of size at least N. This motivates one to generalize this approach to Algebraic-geometric codes (or Goppa codes, henceforth abbreviated as AG-codes) which can have arbitrary block lengths over fixed alphabets, and also have very nice algebraic properties. AG codes are a natural extension of Reed-Solomon codes over algebraic function fields first proposed by Goppa in 1981 [Gop81]. These codes can be viewed as evaluations of regular (or smooth) functions at a set of points on a nice algebraic curve. AG-codes are known to achieve bounds better than the probabilistic constructions over alphabet size $q \geq 49$ [TVZ82, GS95a, GS96b], a quite rare event in combinatorics (also see [Sti91] for a comprehensive treatment). Recent advances have greatly improved the efficiency and explicitness of constructions of AG codes [KAK$^+$01], making this a promising route to approach capacity with better list size and decoding complexity.

The codes in [GR06] are defined over a large alphabet (of size $2^{O(1/\epsilon^4)}$ to get within ϵ of capacity). For codes over alphabet size q for a fixed bounded constant q

(say $q = 2^{12}$), the best general trade-off for error-correction radius vs. rate remains the $(1 - 1/q)(1 - \sqrt{\mathcal{R}})$ bound obtained in [GS99, KV03] for AG codes. Improving this state of affairs provides another motivation for extending the Parvaresh-Vardy approach to AG codes.

We consider designing efficiently list-decodable codes for highly noisy adversarial channels. Situations like this occur naturally in various applications. For example, in complexity theory, list decoding is used in the construction of efficient extractors, in improving average case hardness results [STV01], in obtaining hard-core bits, etc.

We generalize the approach initiated in [PV05] to all AG codes, and define the class of correlated AG codes, based on evaluations of correlated functions from a suitable linear space at points on an algebraic curve. This highlights the generality and promise of the Parvaresh-Vardy approach, and elucidates its salient features in a general setting unencumbered by specifics of a particular code.

We now describe some of our trade-offs for list decoding. For q an even power of a prime, and any integer $m \geq 1$, we present codes with rate $\mathcal{R}$ and list decoding radius approximately $1 - (m\mathcal{R} + 3/\sqrt{q})^{(m+1)/(m+2)}$ over an alphabet of size q^m. (Here m is the number of correlated functions used for the encoding.) For low rates $\mathcal{R}$ and large values of m, this gives an improvement over the trade-off $1 - (\mathcal{R} + 1/\sqrt{q})^{1/2}$ for the usual AG codes (the $m = 1$ case). In particular, for small $\epsilon \to 0$, we can correct up to a fraction $(1 - \epsilon)$ of errors with rate $\Omega(\epsilon/\log(1/\epsilon))$ and alphabet size $2^{O(\log^2(1/\epsilon))}$. Contrast this with the existential result showing that one can list decode to a radius of $(1 - \epsilon)$ with rate $\Omega(\epsilon)$ and alphabet size $O(1/\epsilon^2)$. Our decoding algorithms run in polynomial time assuming a polynomial sized preprocessed representation of the code.

Previously the only polynomial time constructions for decoding up to radius $(1 - \epsilon)$ with alphabet size poly$(1/\epsilon)$ achieved rate $\Omega(\epsilon^2)$ (this follows from the list decoding of AG codes in [GS99]). Our results give the *first codes* with rate better than $\Omega(\epsilon^2)$, say $\Omega(\epsilon^{1.1})$, over an alphabet of size polynomial in $1/\epsilon$. Thus, our result does well simultaneously on both the alphabet size vs. list decoding radius and the rate vs. list decoding radius trade-offs.

Our codes also have a nice *list recovering* property which can be used in concatenation schemes with suitable *constant-sized* inner codes to get the first uniformly constructive binary codes of rate close to ϵ^3 list-decodable up to radius $(1/2 - \epsilon)$

with list size depending only on ϵ and independent of N. (The construction in [GR06] with a similar rate needed construction time of the form $N^{f(\epsilon)}$ instead of the $f(\epsilon)N^{O(1)}$ we achieve, and their list size also depends on N.)

1.1.3 Application to Cryptography

Hash functions are widely-used in cryptography, complexity, and pseudorandomness. For example, the early constructions of extractors are based on hash functions. In the typical cryptographic setting, a hash function is a mapping that maps arbitrary-length strings to fixed length strings. Often we require the length of the output be much smaller than the input. When employed in cryptography, the hash functions are expected to satisfy certain properties. For example, it is *one way* (i.e., it is *easy to compute* and computationally infeasible to invert), it is collision-resistant, etc. A hash function is said to be weakly collision-resistant if given a string x, it is computationally infeasible to find another string y that hashes to the same value. It is said to be collision-resistant if it is computationally infeasible to find *any* two messages that hashes to the same value. Perhaps the main role of a cryptographic hash function is in the provision of message integrity checks and digital signature algorithms. For example, in digital signatures a digital document is signed by hashing the document to a small string using signer's private key.

How can a code be used to construct a good hash function? What properties of codes could be useful? We give some partial answers to these questions. We begin with an analogy to the hardness results obtained from PCPs. Good codes are used to get better PCPs which give better hardness results. For example, a 3-SAT instance is a collection of constraints. By expecting a PCP, i.e., proof in a suitable coded format, the hardness of the problem gets amplified. The basic philosophy is that requiring the proof to be in a coded format, the non-linear constraint satisfaction problem gets difficult. Similarly, a cipher can be viewed as a collection of constraints (possibly non-linear). Here, we are trying to use codes to amplify the computational difficulty. Of course, it is not clear what property of the code can be useful.

A more convincing argument follows from pseudorandomness. Codes are frequently used in pseudorandom constructions and as previously observed cryptographic hash functions seem to require properties similar to pseudorandom objects. Thus it may sound natural to design hash function based on codes. Friedman [Fri86]

used codes with good minimum distance to obtain a family of hash functions that allowed him to construct $O(n \log n)$ size monotone formulae for threshold $k = O(1)$. Our design philosophy of a good hash function is also guided by construction of a code with large distance. However, our motivation follows from differential cryptanalysis. Since we work on improving a specific hash function, namely SHA-1, a more comprehensive account will be given later after we introduce this family of hash functions and mention its weaknesses.

Damgård [Dam89] and Merkle [Mer89] have greatly influenced cryptographic hash function design by defining a hash function in terms of a compression function. A compression function maps a large fixed length block of data to a much shorter fixed length output. Given a large arbitrary large message, the message is first padded, if necessary. It is then broken into an integral number of blocks. These blocks are then processed sequentially to produce a much shorter hash value. Secure Hash Algorithm (SHA) is one such widely used hashing scheme. The basic scheme hashes 512-bit input to a 160 bit output. The SHA family of hash functions is described in [Uni93]. These hash functions consist of two phases: a message expansion phase and a state update transformation, where the state of a cipher is updated in an iterative manner using the expanded messages. Both SHA-0 and SHA-1 have the same state update transformation, but SHA-0 has a simpler message expansion.

In [CJ98] the first theoretical differential collision attack on SHA-0 is described which provably beats the bound obtained from the naive birthday attack. Thereafter several successful attacks have been made on SHA-0, most notably in [CJ98, BC04b, BC04a, WYY05b]. The same strategy has also been effectively used to attack SHA-1 and reduced variants of it, most notably in [BC04a, RO05, MP05, WYY05c]. In the celebrated work [WYY05c], the authors describe a collision attack on full SHA-1 with complexity close to 2^{69} hash operations. This is much smaller than the 2^{80} hash operations that follows from the birthday attack. Therefore it is vital that collision resistant hash functions should be made robust to this line of attack, known as differential attack.

Differential collision attack is the most effective attack against these SHA families of compression functions. In this type of attack, a cleverly designed difference in the messages leads to a zero difference in the output of the block cipher, thus leading to a collision. Unfortunately, in SHA-0 and SHA-1, it is possible to start with a message difference which leads to a small difference in the expanded words.

This in turn allows for a manageable overall differential characteristic to cause a collision.

All the attacks share the same underlying strategy. This basic strategy can briefly be outlined as follows [CJ98]:

1. Approximate the state update transformation (i.e., the cipher) by a linear function.

2. Characterize the kernel (also called differential patterns or disturbance vectors) of the above linear map and show that the kernel has low weight vectors.

3. Show that the message expansion phase can generate these low-weight vectors.

Since the state update function in SHA family is non-linear, roughly the success of the above attack depends on the probability that the non-linear components behave in a linear fashion. This probability depends inverse-exponentially on the weight of the disturbance vectors. To resist the differential attack, it is therefore recommended that the message expansion code should have large minimum distance and should avoid some pattern. In fact, following attacks described in [WYY05b, WYY05c], it is further recommended that the weight of the last 64 words should be large. The authors of [WYY05b, WYY05c] also observe that sometimes the effect of the differential characteristic corresponding to steps 17 to 20 can be annulled with probability 1. Therefore, it would be even better if the weight of the last 60 rounds can be shown to be large. We mention that codes with large distance are known to exist. However, the challenge here is to get a code which is simple and computationally efficient and requires minimal hardware, something similar to SHA-0 or SHA-1 described later.

The authors in [BC04a, RO05, MP05] have been able to generate low-weight differential patterns. These patterns are then used to create collisions or near-collisions in reduced version of SHA-1 with complexity better than the birthday-paradox bound. Extending this further Wang et al. [WYY05c] reports the first attack on the full 80-step SHA-1 with complexity close to 2^{69} hash operations. In there, the authors critically observe that the code not only has small weight codewords (≤ 44, [RO05, WYY05c]) but also that these small weight codewords are even sparser in the last 60 words (for example, [WYY05c] reports a codeword with weight 27 in the last 60 words; also see [JP05a] that proves a lower bound 25 on the

minimum weight in that last 60 words). Wang, Yao and Yao [WYY05a] have further brought the complexity of collision finding in SHA-1 down to 2^{63} hash operations.

These linear codes, a natural generalization of cyclic codes, are known as **quasi-cyclic codes** in the literature. Quasi-cyclic codes[4] have been studied extensively over the last 40 years. (See [TW67, Che92, Lal03, LS05] and the references therein.)

In joint work with Charanjit S. Jutla, we propose a new hash function SHA1-**IME** (**IME** stands for "**I**mproved **M**essage **E**xpansion"). We use the same state update transformation as in SHA-1 or SHA-0. However, we replace the SHA-1 message expansion code by an equally simple code. In [JP05b] we give a computer assisted proof following an elementary linear algebraic argument to conclude that our proposed code has minimum distance at least 82 in just the last 64 words.

Establishing a lower bound on the minimum distance of a quasi-cyclic code is a hard problem and has drawn considerable attention (see [Che92, Lal03]). Unfortunately, when the index (that is, the minimum amount of rotation that leaves the code invariant) is as large as 80 (or even 64), the presently known bound seems computationally infeasible. In general, it is known that computing the minimum weight of an arbitrary linear code is NP-hard (see [Var97]), and that approximating within a constant factor is NP-hard under randomized reductions (see [DMS03]). An interesting approach is taken in [RO05] where they restrict their search by keeping most columns zero. This allows them to find a codeword with low weight for SHA-1; however, they do not give a technique to lower bound the minimum weight of such codes.

The strategy we use to prove lower bounds on such codes is to divide the proof into two main cases. We argue that if there are no zero columns in a codeword (a column in the codeword is the codeword projected on a particular bit position), then on the average each column contributes enough to attain the lower bound. If that is not the case, then starting from an all zero column, the first neighboring non-zero column is actually a codeword in a good code, in the sense that it has large minimum weight, and so on. This neighboring columns then gather enough weight to attain the lower bound. We critically use the fact that the code is a quasi-cyclic code and thus invariant under column shifts. The main challenge is of course to

[4]Interestingly it is known that asymptotically good binary quasi-cyclic exists, a fact which is not yet settled for the cyclic codes.

keep the search space restricted to make the exhaustive search feasible.

In [JP05c] we informally argue that the SHA1-**IME** is resistant to all collision search attacks. We formally argue that at least the present differential collision attacks following [CJ98] local-collision based approach is ineffective as the weight of the code is too large. In particular we prove that arbitrary construction of differential (or disturbance) vector is not possible.

In [JP05c] we show that the cipher can be expressed as a 4-CSP (constraint satisfaction problem where each constraint involves at most 4-variables). The predicates we use are majority and xor. It is known that there is an NP-complete problem [GJ79] involving these predicates. At present all the general-purpose (randomized) algorithms to solve 4-CSP takes 1.4^n or more [Sch99, IT03] (Schoning's algorithm does better in the satisfiable instances, that is in our case of interest). This therefore rules out possibility of collision attacks on SHA1-**IME** using a general purpose algorithm.

Chapter 2

Notation and Preliminaries

The aim of this chapter is to standardize notations which will be used throughout. We also include a few basic facts.

2.1 Notations

For any positive integer l, we denote the set $\{1, \cdots, l\}$ by $[l]$. We will use $\mathbb{F}_p$ to denote a finite field of cardinality p, where p is a prime. We will use $\mathbb{F}_q$ to denote a finite field of size q where $q = p^s$ for some prime p and positive integer s. Sometimes we will simply use $\mathbb{F}$ to denote a finite field, the size of which should be clear from the context. We use $\mathbb{F}_q^*$, or simply $\mathbb{F}^*$, to denote the multiplicative group of the field $\mathbb{F}_q$. For more on finite fields, the readers are referred to [LN94].

2.1.1 Codes

Let $\mathbb{F}_q^N$ be an N-dimensional vector space over $\mathbb{F}_q$. An error correcting code[1] $\mathcal{C}$ of *length* N, *dimension* K, and *alphabet* $\mathbb{F}_q$ is a subset of $\mathbb{F}_q^n$ of cardinality[2] q^K. If the subset $\mathcal{C}$ is a subspace of $\mathbb{F}_q^N$, then it is called a linear code. $\mathcal{C}$ can also be viewed as a mapping from the message space $\mathbb{F}_q^K$ to $\mathbb{F}_q^N$. Whenever the message space of $\mathcal{C}$ is clear from the context, we use $\mathcal{C}(m)$ to denote the image of message m in $\mathbb{F}_q^N$. By a subcode $\mathcal{D}$ of a code $\mathcal{C}$, we mean a proper subset of $\mathcal{C}$ (i.e., $\mathcal{D} \subseteq \mathcal{C}$). On the

[1] The alphabet of an error correcting code need not be a finite field. However, in this thesis we restrict ourselves to codes over finite fields.

[2] We use slightly non-standard notation to enhance readability.

other hand, by a truncated code $\mathcal{C}'$ we mean the same code $\mathcal{C}$ with a few coordinates omitted (or truncated), or equivalently *natural* projection of $\mathcal{C}$ to a smaller vector space. When the number of omitted coordinates is one, the truncated code is also called a *punctured code*.

The rate of a code is defined as $\mathcal{T}(\mathcal{C}) \stackrel{\text{def}}{=} K/N$. We define the (Hamming) distance of a code $\mathcal{C}$ as the quantity

$$d(\mathcal{C}) \stackrel{\text{def}}{=} \min_{u \neq v \in \mathcal{C}} |\{i \mid u_i \neq v_i \text{ where } u = u_1 \cdots u_N, v = v_1, \cdots, v_N\}|.$$

We define the relative distance of the code as the normalized distance, i.e., $\delta(\mathcal{C}) \stackrel{\text{def}}{=} d(\mathcal{C})/N$.

A linear code over an alphabet of size q of dimension K and block length N will be denoted as $[N, K]_q$. Further, if the distance D of the code is known, then the code may also be denoted as $[N, K, D]_q$. For vectors $x, y \in \mathbb{F}^N$, the dot (scalar) product of x and y, denoted $x \cdot y$, is defined to be $\sum_{i=1}^{N} x_i y_i$, where w_i denotes the i^{th} co-ordinate of w.

Given an $[N, K]_q$ code $\mathcal{C}$ over a finite field $\mathbb{F}_q$, its dual code $\mathcal{C}^\perp$ is the set of vectors which are orthogonal to all codewords of $\mathcal{C}$, i.e.,

$$\mathcal{C}^\perp \stackrel{\text{def}}{=} \{u \mid \forall v \in \mathcal{C}, u \cdot v = 0\}.$$

Observe that $\mathcal{C}^\perp$ is a $[N, N-K]_q$ code and $(\mathcal{C}^\perp)^\perp = \mathcal{C}$.

For a more comprehensive treatment on coding theory, the readers are referred to [MS77, vL98].

2.1.2 Low-degree Polynomials as Codes

For any $t \in [n(q-1)]$, let $\mathcal{P}_t$ denote the family of all functions over $\mathbb{F}_q^n$ which are polynomials of total degree at most t (and individual degree at most $q-1$) in n variables. In particular $f \in \mathcal{P}_t$ if there exists coefficients $a_{(e_1,\cdots,e_n)} \in \mathbb{F}_q$, for every $i \in [n]$, $e_i \in \{0, \cdots, q-1\}$, $\sum_{i=1}^{n} e_i \leq t$, such that

$$f(x) = \sum_{\substack{(e_1,\cdots,e_n) \in \{0,\cdots,q-1\}^n \\ 0 \leq \sum_{i=1}^{n} e_i \leq t}} a_{(e_1,\cdots,e_n)} \prod_{i=1}^{n} x_i^{e_i}. \tag{2.1}$$

The codeword corresponding to a function will be the evaluation vector of f.

2.1.3 Finite Derivatives and Tensors

Given a boolean function $f : \mathbb{F}_2^n \to \{-1, 1\}$, we define the *first order "derivative" of f along y* to be the function f_y as follows: $f_y(x) = f(x)f(x + y)$. The higher order derivatives are defined analogously and denoted f_{xy}, f_{xyz} etc. Note that $x, y, z \in \mathbb{F}_2^n$ and that we do not define any multiplicative operation on them. Therefore, the notation is unambiguous and will be employed instead of $f_{x,y}$ and $f_{x,y,z}$ etc. We define the (normalized) distance between two functions to be the probability that they disagree, i.e.,

$$\mathrm{dist}(f, g) \stackrel{\text{def}}{=} \|f - g\| \stackrel{\text{def}}{=} \mathrm{Pr}_{x \in \mathbb{F}_2^n} [\, f(x) \neq g(x) \,].$$

For two boolean functions f and g, we define their convolution as the following function, denoted $f * g$,

$$f * g(x) \quad \stackrel{\text{def}}{=} \quad \mathbb{E}_s f(s)g(x + s).$$

We will use the Kronecker delta function, i.e., δ_ν^μ is 1 iff $\mu = \nu$, and 0 otherwise. For convenience we set $\delta_\nu = \delta_\nu^0$.[3]

If $f(x) = (-1)^{\phi(x)}$ for some function $\phi : \mathbb{F}_2^n \to \mathbb{F}_2$, then we write $Łf(x) = \phi(x)$. A function f is said to be of degree at most d iff its corresponding ϕ is of degree at most d.

Tensors are generalization of matrices in higher dimension. The rank of a given tensor is the number of array indices required to describe such an entry. Thus, a tensor of rank one is just a vector and that of rank two is simply a matrix. In subsection 3.7.3, we use tensors of rank three (to deal with bilinear functions), a three dimensional version of matrices. These may be seen as a family of matrices indexed by some finite set $[n]$. For example, let D a $n \times n \times n$ is a tensor of rank three. Then view $\mathsf{D} = \{\mathsf{D}_i\}_{i \in [n]}$ where each D_i is an $n \times n$ matrix. We abuse notation by letting D^t mean $\{(\mathsf{D}_i)^t\}_{i \in [n]}$, where D^t denotes the transpose of matrix D. For $y \in \{0, 1\}^n$ and D a tensor of rank three, we define $y \cdot \mathsf{D} = \sum_i y_i \mathsf{D}_i$.

[3] We will work over an underlying abelian groups, 0 will always mean the identity element in that abelian group and should be clear from the context.

Later in subsection 3.7.4, we use tensors of even higher rank, precisely rank d, to deal with $(d-1)$-ary linear functions. A tensor D of rank d over $\mathbb{F}_2^n$ should be viewed as a d dimensional generalization of a matrix, i.e., for all $(i_1, \cdots, i_d) \in [n]^d$, $\mathsf{D}_{i_1,\cdots,i_d} \in \mathbb{F}_2$.

2.1.4 Fourier Transform

For a function $f : \mathbb{F}_2^n \to \mathbb{R}$, we use $\hat{f} : \mathbb{F}_2^n \to \mathbb{R}$ to denote its Fourier transform. We will use fourier transform over the additive group of $\mathbb{F}_2^n$ unless otherwise stated. We will use standard facts from Fourier analysis, namely the following:

1. (Fourier transform) $\hat{f}(\alpha) = \mathbb{E}_x f(x)\chi_\alpha(x)$.
2. (Fourier inversion) $f(x) = \sum_\alpha \hat{f}(\alpha)\chi_\alpha(x)$.
3. (Parseval/Plancharel equality) $\sum_\alpha \hat{f}_\alpha^2 = 1$.
4. (Orthogonality) $\mathbb{E}_x \chi_\alpha(x)\chi_\beta(x) = \delta_\beta^\alpha$.

Chapter 3

Local Testability of Reed-Muller Codes

3.1 Introduction

The term locally-testable codes (LTC) has its origin in the seminal paper of Rubinfeld and Sudan [RS96] (also in [FS95], and the same idea was present in Arora's and Spielman's PhD thesis under different names). Informally a code is locally testable (*with one-sided error*) *iff* given oracle (black box) access to a given purported codeword w there is an efficient randomized algorithm that queries w in a small number of places, accepts with probability one if w is a valid codeword, and rejects with high probability if w is far from being a valid codeword. Formally,

Definition 3.1.1 *A code $\mathcal{C}$ of block length n is said to be (r, δ, σ)-locally testable if there exists an efficient randomized algorithm $\mathcal{A}$, a polynomial $p(\cdot)$, a function $F : \{0,1\}^{p(n)} \to [n]^r$ with the property that*

- *For every $w \in \mathcal{C} \subseteq \mathbb{F}_q^n$, $\Pr_{\rho \in \{0,1\}^{p(n)}} [\, \mathcal{A}(\{w_i | i \in F(\rho)\}) = 1 \,] = 1$.*
- *For every string y such that $d(y, \mathcal{C}) \geq \delta$*

$$\Pr_{\rho \in \{0,1\}^{p(n)}} [\, \mathcal{A}(\{w_i | i \in F(\rho)\}) = 1 \,] \leq \sigma$$

We quickly recall the definition of the Generalized (Primitive) Reed-Muller code as described in [DGM70, AJK98, DK00].

Definition 3.1.2 *Let $V = \mathbb{F}_q^n$ be the vector space of n-tuples, for $n \geq 1$, over the field $\mathbb{F}_q$. For any k such that $0 \leq k \leq n(q-1)$, the k^{th} order Generalized Reed-Muller code* $\mathrm{GRM}_q(k,n)$ *is the subspace of $\mathbb{F}_q^{|V|}$ (with the basis as the characteristic functions of vectors in V) of all n-variable polynomial functions (reduced modulo $x_i^q - x_i$) of degree at most k.*

The local testability of `Generalized Reed-Muller codes` (henceforth abbreviated as GRM) is better known as the low-degree testing problem. Informally, the local testability of GRM codes can be stated as follows. Given a black box function $f : \mathbb{F}_q^n \to \mathbb{F}_q$, determine whether f is (sufficiently) close, in the Hamming metric, to a low-degree (say, degree at most k) polynomial. In this chapter, we present a low degree test for multivariate polynomials over any prime field $\mathbb{F}_p$.

3.1.1 Background and Context

A *low-degree tester* is a probabilistic algorithm which, given a degree parameter t and oracle access to a function f on n arguments (which take values from some finite field $\mathbb{F}$), has the following behavior. If f is the evaluation of a polynomial on n variables with total degree at most t, then the low-degree tester must accept with probability one. On the other hand, if f is "far" from being the evaluation of some polynomial on n variables with degree at most t, then the tester must reject with constant probability. The tester can query the function f to obtain the evaluation of f at any point. However, the goal of a tester is to accomplish its task by using as few probes as possible.

Low-degree testers play an important part in the construction of Probabilistically Checkable Proofs (or PCPs). In fact, different parameters of low degree testers (for example, the number of probes and the amount of randomness used) directly affect the parameters of the corresponding PCPs as well as various inapproximability results obtained from such PCPs ([FGL$^+$91, ALM$^+$92]). Low-degree testers also form the core of the proof of MIP = NEXPTIME in [BFL91].

3.1.2 Previous Low-degree Testers

The study of low degree testing (along with *self-correction*) dates back to the work of Blum, Luby and Rubinfeld ([BLR93]), where an algorithm was required to test whether a given function is linear. The approach in [BLR93] later naturally extended

to yield testers for low-degree polynomials (but over fields larger than the total degree). Roughly, the idea is to project the given function on to a random line and then test if the projected univariate polynomial has low degree. Specifically, for a purported degree t function $f : \mathbb{F}_q^n \to \mathbb{F}_q$, the test works as follows. Pick vectors y and b from $\mathbb{F}_q^n$ (uniformly at random), and distinct $s_1, \cdots, s_{t+1}$ from $\mathbb{F}_q$ arbitrarily. Query the oracle representing f at the $t+1$ points $b + s_i y$ and extrapolate to a degree t polynomial $P_{b,y}$ in one variable s. Now test for a random $s \in \mathbb{F}_p$ whether

$$P_{b,y}(s) = f(b + sy)$$

(for details see [RS96], [FS95]). Similar ideas are also employed to test whether a given function is a low-degree polynomial in each of its variables (see [FGL$^+$91, BFLS91, AS92]). Note that this approach does not work when the field size is smaller than the total degree, as $x^q = x$ in $\mathbb{F}_q$.

Alon et al. give a tester over field $\mathbb{F}_2$ for any degree up to the number of inputs to the function (i.e., for any non-trivial degree) [AKK$^+$03]. In other words, their work shows that Reed-Muller codes are locally testable. Under the coding theoretic interpretation, their tester picks a random low-weight codeword from the dual code and checks if it is orthogonal to the input vector.

Specifically their test works as follows: Given a function $f : \{0,1\}^n \to \{0,1\}$, to test if the given function f has degree at most t, pick $(t+1)$-vectors $y_1, \cdots, y_{t+1} \in \{0,1\}^n$ and test if

$$\sum_{\emptyset \neq S \subseteq [t+1]} f\left(\sum_{i \in S} y_i\right) = 0.$$

As we show later, the test in [RS96] above can also be obtained by using this coding theoretic interpretation.

3.1.3 Overview

It is easier to define our tester over $\mathbb{F}_3$. To test if f has degree at most t, set $k = \lceil \frac{t+1}{2} \rceil$, and let $i = (t+1) \pmod 2$. Pick k-vectors $y_1, \cdots, y_k$ and b from $\mathbb{F}_3^n$,

and test if[1]

$$\sum_{c\in\mathbb{F}_3^k; c=(c_1,\cdots,c_k)} c_1^i f(b+\sum_{j=1}^{k} c_j y_j) = 0.$$

We prove that a polynomial of degree at most t always passes the test, whereas a polynomial of degree greater than t gets caught with non-negligible probability α. To obtain a constant rejection probability we repeat the test $\Theta(1/\alpha)$ times.

As in [RS96] there are two main parts to the proof. The first step is coming up with an *exact characterization* for functions that have low degree. Following [AKK$^+$03], it is best to view low degree polynomials over $\mathbb{F}_q$ as the GRM code. As GRM is a linear code, a function is of low degree if and only if it is orthogonal to every codeword in the dual of the corresponding GRM code. The second step of the proof entails showing that the characterization is a *robust characterization*, that is, the following natural tester is indeed a local tester. Pick one of the dual low-weight codewords uniformly at random and check if it is orthogonal to the given function.

The analysis of our test follows a similar general structure developed in [RS96] and borrows techniques from [RS96, AKK$^+$03]. The presence of a doubly transitive group suffices for the analysis given in [RS96]. Essentially we show that the presence of a doubly transitive group acting on the coordinates of the dual code does indeed allow us to localize the test. However, this gives a weaker result. We use techniques developed in [AKK$^+$03] for better results, although the adoption is not immediate. Apart from the obvious difficulty of proving step two, the proof is further complicated by the fact that to obtain a good tester (i.e., one which makes as few queries as possible), we need a sub-collection of the dual GRM code in which each vector has low weight (and it generates the dual code).

Since it is well known that the dual of a GRM code is a GRM code (with different parameters), to obtain a collection of codewords (with low weight) that generate the dual of a GRM code it is enough to do so for the GRM code itself. We present a new basis of GRM codes over prime fields that in general differs from the minimum weight basis obtained in [DK00]. Our basis has a clean geometric structure in terms of *flats* [AJK98], and unions of parallel flats (but with different weights assigned to different parallel flats)[2]. This equivalent polynomial and geometric representation plays a pivotal role in proving step two (and with almost

[1]For notational convenience we use $0^0 = 1$.

[2]The natural basis given in [DGM70, DK00] assigns the same weight to each parallel flat.

optimal query complexity). In Section 3.3 we present an exact characterization for low-degree polynomials over prime fields. In Section 3.4 we extend the characterization over general fields and sketch an alternate proof to the theorem of Kaufman and Ron[KR04]. In Section 3.5 we present a tester and prove its correctness. In Section 3.6 we give a lower bound (essentially due to Alon, Krivelevich, Newman and Szegedy [AKNS99]) on the number of queries to test Reed-Muller codes to demonstrate that our result is very close to the optimal. In Section 3.7 we present our observation on the inverse theorem for the Gowers norm.

3.2 Preliminaries

For any two functions $f, g : \mathbb{F}_q^n \to \mathbb{F}_q$, the relative distance $\delta(f,g) \in [0,1]$ between f and g is defined as $\delta(f,g) \stackrel{def}{=} \Pr_{x \in \mathbb{F}_q^n}[\, f(x) \neq g(x)\,]$. For a function g and a family of functions F (defined over the same domain and range), we say g is ϵ-`close` to F, for some $0 < \epsilon < 1$, if, there exists an $f \in F$, where $\delta(f,g) \leq \epsilon$. Otherwise it is ϵ-`far` from F.

Recall that a one sided testing algorithm (`one-sided tester`) for $\mathcal{P}_t$ is a probabilistic algorithm that is given query access to a function f and a distance parameter ϵ, $0 < \epsilon < 1$. If $f \in \mathcal{P}_t$, then the tester should always accept f (perfect completeness), and if f is ϵ-`far` from $\mathcal{P}_t$, then with probability at least $\frac{1}{2}$ the tester should reject f (a two-sided tester may be defined analogously).

To motivate the next notation which we will use frequently, we give a definition.

Definition 3.2.1 *A k-flat ($k \geq 0$)[3] in $\mathbb{F}_p^n$ is a k-dimensional affine subspace. Let $y_1, \cdots, y_k \in \mathbb{F}_p^n$ be linearly independent vectors and $b \in \mathbb{F}_p^n$ be a point. Then the subset $L = \{\sum_{i=1}^k c_i y_i + b \mid \forall i \in [k]\ c_i \in \mathbb{F}_p\}$ is a k-dimensional flat. We will say that L is generated by $y_1, \cdots, y_k$ at b. The incidence vector of the points in a given k-flat will be referred to as the codeword corresponding to the given k-flat.*

Given a function $f : \mathbb{F}_p^n \to \mathbb{F}_p$, for $y_1, \cdots, y_l, b \in \mathbb{F}_p^n$ we define

$$T_f(y_1, \cdots, y_l, b) \stackrel{def}{=} \sum_{c=(c_1,\cdots,c_l) \in \mathbb{F}_p^l} f\left(b + \sum_{i \in [l]} c_i y_i\right), \tag{3.1}$$

[3] A zero-dimensional flat is just a point.

which is the sum of the evaluations of function f over an l-flat generated by $y_1, \cdots, y_l$ at b. Alternatively, this can also be interpreted as the dot product of the codeword corresponding to the l-flat generated by $y_1, \cdots, y_l$ at b and that corresponding to the function f (see Observation 3.3.5).

While k-flats are well-known, we define a new geometric object, called a pseudoflat. A k-pseudoflat is a union of $(p-1)$ parallel $(k-1)$-flats. Also, a k-pseudoflat can have different exponents ranging from 1 to[4] $(p-2)$. We stress that the point set of a k-pseudoflat remains the same irrespective of its exponent. It is the value assigned to a point that changes with the exponents.

Definition 3.2.2 *Let $L_1, L_2, \cdots, L_{p-1}$ be parallel $(k-1)$-flats ($k \geq 1$), such that for some $y \in \mathbb{F}_p^n$ and all $t \in [p-2]$, $L_{t+1} = y + L_t$.*[5] *We define the* `points` *of k-pseudoflat L with any exponent r ($1 \leq r \leq p-2$) to be the union of the set of points L_1 to L_{p-1}. Also, let I_j be the incidence vector of L_j for $j \in [p-1]$. Then the* `evaluation vector` *of this k-*`pseudoflat with exponent` *r is defined to be $\sum_{j=1}^{p-1} j^r I_j$. The evaluation vector of a k-pseudoflat with exponent r will be referred as the codeword corresponding to the given k-pseudoflat with exponent r.*

Let L be a k-pseudoflat with exponent r. Also, for $j \in [p-1]$, let L_j be the $(k-1)$-flat generated by $y_1, \cdots, y_{k-1}$ at $b + j \cdot y$, where $y_1, \cdots, y_{k-1}$ are linearly independent. Then we say that L, a k-pseudoflat with exponent r, is generated by $y, y_1, \cdots, y_{k-1}$ at b exponentiated along y.

Given a function $f : \mathbb{F}_p^n \to \mathbb{F}_p$, for $y_1, \cdots, y_l, b \in \mathbb{F}_p^n$, for all $i \in [p-2]$, we similarly define

$$T_f^i(y_1, \cdots, y_l, b) \stackrel{def}{=} \sum_{c=(c_1,\cdots,c_l)\in\mathbb{F}_p^l} c_1^i \cdot f\left(b + \sum_{j\in[l]} c_j y_j\right). \tag{3.2}$$

The above can also be interpreted similarly as the dot product of the codeword corresponding to the l-pseudoflat with exponent i generated by $y_1, \cdots, y_l$ at b and the codeword corresponding to the function f (see Observation 3.3.9). With a slight abuse of notation[6] we will use $T_f^0(y_1, \cdots, y_l, b)$ to denote $T_f(y_1, \cdots, y_l, b)$.

[4] With slight abuse, a k-pseudoflat with exponent zero corresponds to a flat.

[5] For a set $S \subseteq \mathbb{F}_p^n$ and $y \in \mathbb{F}_p^n$, we define naturally $y + S \stackrel{\text{def}}{=} \{x + y | x \in S\}$.

[6] We set $0^0 = 1$, for notational convenience.

3.2.1 Facts from Finite Fields

In this section we spell out some facts from finite fields which will be used later. Recall that we denote the multiplicative group of $\mathbb{F}_q$ by $\mathbb{F}_q^*$. We begin with a simple lemma.

Lemma 3.2.3 *For any $t \in [q-1]$, $\sum_{a \in \mathbb{F}_q} a^t \neq 0$ if and only if $t = q-1$.*

Proof: First note that $\sum_{a \in \mathbb{F}_q} a^t = \sum_{a \in \mathbb{F}_q^*} a^t$. Observing that for any $a \in \mathbb{F}_q^*$, $a^{q-1} = 1$, it follows that $\sum_{a \in \mathbb{F}_q^*} a^{q-1} = \sum_{a \in \mathbb{F}_q^*} 1 = -1 \neq 0$.

Next we show that for all $t \neq q-1$, $\sum_{a \in \mathbb{F}_q^*} a^t = 0$. Let α be a generator of $\mathbb{F}_q^*$. The sum can be re-written as $\sum_{i=0}^{q-2} \alpha^{it} = \frac{\alpha^{t(q-1)}-1}{\alpha^t - 1}$. The denominator is non-zero for $t \neq q-1$ and thus, the fraction is well defined. The proof is complete by noting that $\alpha^{t(q-1)} = 1$. ■

This immediately implies the following lemma.

Lemma 3.2.4 *Let $t_1, \cdots, t_l \in [q-1]$. Then*

$$\sum_{(c_1,\cdots,c_l) \in \mathbb{F}_q^l} c_1^{t_1} c_2^{t_2} \cdots c_l^{t_l} \neq 0 \text{ if and only if } t_1 = t_2 = \cdots = t_l = q-1. \tag{3.3}$$

Proof: Note that the left hand side can be rewritten as $\prod_{i \in [l]} \left(\sum_{c_i \in \mathbb{F}_q} c_i^{t_i} \right)$. ■

We will need to transform products of variables to powers of linear functions in those variables. With this motivation, we present the following identity.

Lemma 3.2.5 *For each k, s.t. $0 < k \leq (p-1)$ there exists $c_k \in \mathbb{F}_p^*$ such that*

$$c_k \prod_{i=1}^{k} x_i = \sum_{i=1}^{k} (-1)^{k-i} S_i \quad \text{where} \quad S_i = \sum_{\emptyset \neq I \subseteq [k]; |I| = i} \left(\sum_{j \in I} x_j \right)^k. \tag{3.4}$$

Proof: Consider the right hand side of the Equation 3.4. Note that all the monomials are of degree exactly k. Also note that $\prod_{i=1}^{k} x_i$ appears only in the S_k and nowhere else. Now consider any other monomial of degree k that has a support of size j, where $0 < j < k$. Further note that the coefficient of any such monomial in the expansion of $(\sum_{j \in I} x_j)^k$ is the same and non-zero. Therefore, summing up

the number of times it appears (along with the $(-1)^{k-i}$ factor) in each S_i is enough which is just

$$1 - \binom{k-j}{k-j-1} + \binom{k-j}{k-j-2} + \cdots + (-1)^{k-j}\binom{k-j}{k-j-(k-j)} = (1-1)^{k-j} = 0.$$

Moreover, it is clear that $c_k = k! \pmod p$ and $c_k \neq 0$ for the choice of k. ■

3.3 Characterization of Low-degree Polynomials over Prime Fields

In this section we present an exact characterization for the family $\mathcal{P}_t$ over prime fields. Specifically we prove the following:

Theorem 3.3.1 *Let $t = (p-1)\cdot k + R$. (Note $0 \leq R \leq p-2$.) Let $r = p-2-R$. Then a function f belongs to $\mathcal{P}_t$, if and only if* `for every` $y_1, \cdots, y_{k+1}, b \in \mathbb{F}_p^n$, *we have*

$$T_f(y_1, \cdots, y_{k+1}, b) = 0 \quad \text{if } r = 0; \tag{3.5}$$

$$T_f^r(y_1, \cdots, y_{k+1}, b) = 0 \quad \text{otherwise.} \tag{3.6}$$

As mentioned previously, a characterization for the family $\mathcal{P}_t$ implies a characterization for $\mathrm{GRM}_p(t,n)$ and vice versa. It turns out that it is easier to characterize $\mathcal{P}_t$ when viewed as $\mathrm{GRM}_p(t,n)$. Therefore our goal is to determine whether a given word belongs to the GRM code. Since we deal with a linear code, a simple strategy will then be to check whether the given word is orthogonal to all the codewords in the dual code. Though this yields a characterization, this is computationally inefficient. Note however that the dot product is linear in its input. Therefore checking orthogonality with a basis of the dual code suffices. To make it computationally efficient, we look for a basis with small weights. The above theorem essentially is a clever restatement of this idea.

We recall the following useful lemma that can be found in corollary 5.26 of [AJK98].

Lemma 3.3.2 $\mathrm{GRM}_q(k,n)$ *is a linear code with block length q^n and minimum distance $(R+1)q^Q$ where R is the remainder and Q the quotient resulting from dividing $(q-1)\cdot n - k$ by $(q-1)$. Further* $\mathrm{GRM}_q(k,n)^\perp = \mathrm{GRM}_q((q-1)\cdot n - k - 1, n)$.

Proof: We only prove $\mathrm{GRM}_q(k,n)^\perp = \mathrm{GRM}_q((q-1)n-k-1,n)$ for which we give an easy proof. Denote $\mathcal{C} \stackrel{\text{def}}{=} \mathrm{GRM}_q(k,n)$ and $\mathcal{D} \stackrel{\text{def}}{=} \mathrm{GRM}_q((q-1)n-k-1,n)$. We first prove $\mathcal{D} \subseteq \mathcal{C}^\perp$. This is easy as given any codeword in $u \in \mathcal{C}$ and $v \in \mathcal{D}$, the total degree of $u \cdot v$ is at most $(q-1)n-1$. Hence by Lemma 3.2.4, $u \cdot v = 0$.

Note that any function $f : \mathbb{F}_q^n \to \mathbb{F}$ can be represented by a poly of degree at most $n(q-1)$. Now to prove that $\mathcal{D}$ is exactly the set of polynomials that are orthogonal to $\mathrm{GRM}_q(k,n)$, we show that any polynomial not in $\mathcal{D}$ is not orthogonal to at least one polynomial in $\mathrm{GRM}_q(k,n)$. Let u be any polynomial of degree at least $(q-1)n-k$. Let m be a monomial in u that has maximal degree, but otherwise arbitrary. Denote $m(x) = \prod_i x_i^{e_i}$. Then define $g(x) = \prod_i x_i^{q-1-e_i}$. Clearly degree$(g) \leq k$. Therefore, by Lemma 3.2.4 and linearity of dot product, $f \cdot g \neq 0$. This completes the proof. ■

Since the dual of a GRM code is again a GRM (of appropriate order), we therefore need the generators of GRM code (of arbitrary order). We first establish that flats and pseudoflats (of suitable dimension and exponent) indeed generate the Generalized Reed-Muller code (of desired order). We then end the section with a proof of Theorem 3.3.1 and a few remarks.

We begin with few simple observations about flats. Note that an l-flat L is the intersection of $(n-l)$ hyperplanes in general position. Equivalently, it consists of all points v that satisfy $(n-l)$ linear equations over $\mathbb{F}_p$ (i.e., one equation for each hyperplane): $\forall i \in [n-l] \quad \sum_{j=1}^n c_{ij}x_j = b_i$ where c_{ij}, b_i defines the i^{th} hyperplane (i.e., v satisfies $\sum_{j=1}^n c_{ij}v_j = b_i$). General position means that the matrix $\{c_{ij}\}$ has rank $(n-l)$. Note that then the incidence vector of L can be written as

$$\prod_{i=1}^{n-l}\left(1-\left(\sum_{j=1}^{n} c_{ij}x_j - b_i\right)^{p-1}\right) = \begin{cases} 1 & \text{if } (v_1,\cdots,v_l) \in L \\ 0 & \text{otherwise} \end{cases} \tag{3.7}$$

We record a lemma here that will be used later in this section. We leave the proof as a straightforward exercise.

Lemma 3.3.3 *For $l \geq k$, the incidence vector of any l-flat is a linear sum of the incidence vectors of k-flats.*

As mentioned previously, we give an explicit basis for $\mathrm{GRM}_p(r, n)$. For the special case of $p = 3$, our basis coincides with the min-weight basis given in [DK00].[7] However, in general, our basis differs from the min-weight basis provided in [DK00].

The following Proposition shows that the incidence vectors of flats form a basis for the Generalized Reed-Muller code of orders which are multiples of $(p-1)$.

Proposition 3.3.4 $\mathrm{GRM}_p((p-1)(n-l), n)$ *is generated by the incidence vectors of the* l*-flats.*

Proof: We first show that the incidence vectors of the l-flats are in $\mathrm{GRM}_p((p-1)(n-l), n)$. Recall that L is the intersection of $(n-l)$ independent hyperplanes. Therefore using Equation 3.7, L can be represented by a polynomial of degree at most $(n-l)(p-1)$ in $x_1, \cdots, x_n$. Therefore the incidence vectors of l-flats are in $\mathrm{GRM}_p((p-1)(n-l), n)$.

We prove that $\mathrm{GRM}_p((p-1)(n-l), n)$ is generated by l-flats by induction on $n-l$. When $n-l = 0$, the code consists of constants, which is clearly generated by n-flats, i.e., the whole space.

To prove for an arbitrary $(n-l) > 0$, we show that any monomial of total degree $d \leq (p-1)(n-l)$ can be written as a linear sum of the incidence vectors of l-flats. Let the monomial be $x_1^{e_1} \cdots x_t^{e_t}$. Rewrite the monomials as $\underbrace{x_1 \cdots x_1}_{e_1 \text{ times}} \cdots \underbrace{x_t \cdots x_t}_{e_t \text{ times}}$. Group into products of $(p-1)$ (not necessarily distinct) variable as much as possible. Rewrite each group using Equation 3.4 setting $k = (p-1)$. For any incomplete group of size d', use the same equation by setting the last $(p-1-d')$ variables to the constant 1. After expansion, the monomial can be seen to be a sum of product of at most $(n-l)$ degree $(p-1)^{\text{th}}$ powered linear terms. We can add to it a polynomial of degree at most $(p-1)(n-l-1)$ so as to represent the resulting polynomial as a sum of polynomials, each polynomial as in Equation 3.7. Each such non-zero polynomial is generated by a t flat, $t \geq l$. By induction, the polynomial we added is generated by $(l+1)$ flats. Thus, by Lemma 3.3.3 our given monomial is generated by l-flats. ■

This leads to the following observation:

[7] The equations of the hyperplanes are slightly different in our case; nonetheless, both of them define the same basis generated by the min-weight codewords.

Observation 3.3.5 *Consider an l-flat generated by $y_1, \cdots, y_l$ at b. Denote the incidence vector of this flat by I. Then the right hand side of Equation 3.1 may be identified as $I \cdot f$, where I and f denote the vector corresponding to respective codewords and $\cdot$ is the dot (scalar) product.*

To generate generalized Reed-Muller code of any arbitrary order, we need pseudoflats. Note that the points in a k-pseudoflat may alternatively be viewed as the space given by union of intersections of $(n-k-1)$ hyperplanes, where the union is parameterized by another hyperplane that does not take one particular value. Concretely, it is the set of points v which satisfy the following constraints over $\mathbb{F}_p$:

$$\forall i \in [n-k-1] \sum_{j=1}^{n} c_{ij}x_j = b_i; \text{ and } \quad \sum_{j=1}^{n} c_{n-k,j}x_j \neq b_{n-k}.$$

Thus the values taken by the points of a k-pseudoflat with exponent r is given by the polynomial

$$\prod_{i=1}^{n-k-1} \left(1 - \left(\sum_{j=1}^{n} c_{ij}x_j - b_i\right)^{(p-1)}\right) \cdot \left(\sum_{j=1}^{n} c_{n-k,j}x_j - b_{n-k}\right)^{r} \tag{3.8}$$

Remark 3.3.6 *Note the difference between Equation 3.8 and the basis polynomial in [DK00] that (along with the action of the affine general linear group) yields the min-weight codewords:*

$$h(x_1, \cdots, x_m) = \prod_{i=1}^{k-1} \left(1 - (x_i - w_i)^{(p-1)}\right) \prod_{j=1}^{r} (x_k - u_j),$$

where $w_1, \cdots, w_{k-1}, u_1, \cdots, u_r \in \mathbb{F}_p$.

The next lemma shows that the code generated by the incidence vectors of l-flats is a subcode of the code generated by the evaluation vectors of l-pseudoflats with exponent r.

Claim 3.3.7 *The evaluation vectors of l-pseudoflats ($l \geq 1$) with exponent r ($r \in [p-2]$) generate a code containing the incidence vectors of l-flats.*

Proof: Let W be the incidence vector of an l-flat generated by $y_1, \cdots, y_l$ at b. Clearly $W = \langle 1, \cdots, 1 \rangle$, where the i^{th} $(i \in [p-1] \cup \{0\})$ coordinate denotes the values taken by the characteristic functions of $(l-1)$-flats generated by $y_2, \cdots, y_l$ at $b+i \cdot y_1$.[8] Let this denote the standard basis. Let L_j be a pseudoflat with exponent r generated by $y_1, \cdots, y_l$ exponentiated along y_1 at $b+j \cdot y_1$, for each $j \in \mathbb{F}_p$, and let V_j be the corresponding evaluation vector. By Definition 3.2.2, V_j assign a value i^r to the $(l-1)$-flat generated by $y_2, \cdots, y_l$ at $b+(j+i)y$. Rewriting them in the standard basis yields that $V_j = \langle (p-j)^r, (p-j+1)^r, \cdots, (p-j+i)^r, \cdots, (p-j-1)^r \rangle \in \mathbb{F}_p^p$. Let λ_j denote p variables for $t = 0, 1, \cdots, (p-1)$, each taking values in $\mathbb{F}_p$. Then a solution to the following system of equations

$$\forall i \in [p-1] \cup \{0\} \quad 1 = \sum_{j \in \mathbb{F}_p} \lambda_j (i-j)^r$$

implies that $W = \sum_{j=0}^{p-1} \lambda_j V_j$, which suffices to establish the claim. Consider the identity

$$1 = (-1) \sum_{j \in \mathbb{F}_p} (j+i)^r j^{p-1-r}$$

which may be verified by expanding and applying Lemma 3.2.3. Setting λ_j to $(-1)(-j)^{p-1-r}$ establishes the claim. ■

The next Proposition complements Proposition 3.3.4. Together they say that by choosing dimension and exponent appropriately, Generalized Reed-Muller code of any given order can be generated. This gives an equivalent representation of Generalized Reed-Muller code. An exact characterization then follows from this alternate representation.

Proposition 3.3.8 *For every $r \in [p-2]$, the linear code generated by the evaluation vectors of l-pseudoflats with exponent r is equivalent to* $\mathrm{GRM}_p((p-1)(n-l)+r, n)$.

Proof: For the forward direction, consider an l-pseudoflat L with exponent r. Its evaluation vector is given by an equation similar to Equation 3.8. Thus the codeword corresponding to the evaluation vector of this flat can be represented by a polynomial of degree at most $(p-1)(n-l)+r$. This completes the forward direction.

[8] Recall that a l-pseudoflat (as well as a flat) assigns the same value to all points in the same $(l-1)$-flat.

To prove the other direction, we restrict our attention to monomials of degree at least $(p-1)(n-l)+1$ and show that these monomials are generated by l-pseudoflats with exponent r. Since monomials of degree at most $(p-1)(n-l)$ is generated by l-flats, Claim 3.3.7 will establish the Proposition. Now consider any such monomial. Let the degree of the monomial be $(p-1)(n-l)+r'$ $(1 \leq r' \leq r)$. Rewrite it as in Proposition 3.3.4. Since the degree of the monomial is $(p-1)(n-l)+r'$, we will be left with an incomplete group of degree r'. We make any incomplete group complete by adding 1's (as necessary) to the product. We then use Lemma 3.2.5 to rewrite each (complete) group as a linear sum of r^{th} powered terms. After expansion, the monomial can be seen to be a sum of product of at most $(n-l)$ degree $(p-1)^{th}$ powered linear terms and a r^{th} powered linear terms. Each such polynomial is generated either by an l-pseudoflat with exponent r or an l-flat. Claim 3.3.7 completes the proof. ■

The following is analogous to Observation 3.3.5.

Observation 3.3.9 *Consider an l-pseudoflat with exponent r, generated by $y_1, \cdots, y_l$ at b exponentiated along y_1. Let E be the evaluation vector of this pseudoflat with exponent r. Then the right hand side of Equation 3.2 may be interpreted as $E \cdot f$.*

Now we prove the exact characterization.

Proof of Theorem 3.3.1: The proof directly follows from Lemma 3.3.2, Proposition 3.3.4, Proposition 3.3.8 and Observation 3.3.5 and Observation 3.3.9. Indeed by Observation 3.3.5 and Observation 3.3.9, Equations 3.5 and 3.6 are essentially tests to determine whether the dot product of the function with every vector in the dual space of $\mathrm{GRM}(t,n)$ evaluates to zero. ■

Remark 3.3.10 *One can obtain an alternate characterization from Remark 3.3.6 which we state here without proof.*

Let $t = (p-1)\cdot k + R$ (note $0 < R \leq (p-2)$). Let $r = (p-1) - R - 1$. Let $W \subseteq \mathbb{F}_p$ with $|W| = r$. Define the polynomial $g(x) \stackrel{def}{=} \prod_{\alpha \in W}(x-\alpha)$ if W is non-empty; and $g(x) = 1$ otherwise. Then a function belong to $\mathcal{P}_t$ if and only if for every $y_1, \cdots, y_{k+1}, b \in \mathbb{F}_p^n$, we have

$$\sum_{c_1 \in \mathbb{F}_p \setminus W} g(c_1) \sum_{(c_2,\cdots,c_{k+1}) \in \mathbb{F}_p^k} f\left(b + \sum_{i=1}^{k+1} c_i \cdot y_i\right) = 0.$$

Moreover, this characterization can also be extended to certain degrees for more general fields, i.e., $\mathbb{F}_{p^s}$ (see the next remark).

Remark 3.3.11 *The exact characterization of low degree polynomials as claimed in [FS95] may be proved using duality. Note that their proof works as long as the dual code has a min-weight basis (see [DK00]). Suppose that the polynomial has degree $d \leq q - q/p - 1$, then the dual of $\mathrm{GRM}_q(d, n)$ is $\mathrm{GRM}_q((q-1)n - d - 1, n)$ and therefore has a min-weight basis. Note that then the dual code has min-weight $(d+1)$. Therefore, assuming the minimum weight codewords constitute a basis, any $d+1$ evaluations of the original polynomial on a line are dependent and vice-versa. We leave the details as an exercise for the interested readers.*

3.4 Characterization of Low-degree Polynomials over General Finite Fields

In an attempt to generalize our result from previous section to more general fields, we obtain an exact characterization of low-degree polynomials over general finite fields[9][JPR04]. This provides an alternate proof to the result of Kaufman and Ron [KR04] mentioned earlier. Specifically the result says that a given polynomial is of degree at most t *if and only if* the restriction of the polynomial to every affine subspace of dimension $\lceil \frac{t+1}{q-q/p} \rceil$ (and higher) is of degree at most t.

The result of this section was first proved in [KR04]. Their proof generalizes a proof presented in [FS95]. Here we present an alternative proof to their result, which also extends our result presented in the previous section. In this subsection we characterize low-degree polynomials over general field $\mathbb{F}_q$ of characteristic p. We approach this problem the same way we do in the previous chapter. We set up a characterization via the dual code. That is, a given function is of degree at most t, if it is orthogonal to all the codewords in the dual code of $\mathrm{GRM}_q(t, n)$. The orthogonality test can also be seen as a weighted sum of the evaluation of the given function on a small subset of points. We then show that each of these weighted sum can be represented as a linear sum of the weighted sum of the evaluation of the given function over objects which we call *funny-flats*. In other words, these funny-flats

[9] We mention here that this can further be extended to a robust characterization using techniques we develop for prime fields.

generate the dual code of $\mathrm{GRM}_q(t,n)$.

Denote $d \stackrel{\text{def}}{=} \lceil \frac{t+1}{q-q/p} \rceil$. We show that

Theorem 3.4.1 *A function $f : \mathbb{F}_q^n \to \mathbb{F}_q$ belongs to $\mathcal{P}_t$ if and only if for every $y_1, \cdots, y_d, b \in \mathbb{F}_q^n$, every $(q-1) \geq r_1, \cdots, r_d \geq 0$ such that $\sum_{i=1}^{d}(q-1-r_i) > t$, it holds that*

$$\sum_{a=\langle a_1,\cdots,a_d\rangle \in \mathbb{F}_q^d} \prod_{i=1}^{d} a_i^{r_i} f\left(b + \sum_{j=1}^{d} a_j y_j\right) = 0. \tag{3.9}$$

Since the set of monomials given by $\{\prod_{i=1}^{d} x_i^{r_i} |$ for all $i \in [d]$ $r_i \in \mathbb{Z}/q\mathbb{Z}$ and $\sum_{i=1}^{d}(q-1-r_i) > t\}$ generate the code $\mathrm{GRM}_q(d(q-1)-t-1, d)$, using duality (Lemma 3.3.2) we immediately get the following theorem as a corollary which appears in [KR04].

Theorem 3.4.2 *(Kaufman-Ron [KR04]) Let $f : \mathbb{F}_q^n \to \mathbb{F}_q$. Then f belongs to $\mathcal{P}_t$ iff the restriction of f to every affine subspace $\mathcal{H}$ of dimension $d = \lceil \frac{t+1}{q-q/p} \rceil$ is a polynomial of degree at most t.*

We start with a definition.

Definition 3.4.3 *Given t and $n \geq d$, a set of non-negative exponents $r_1, \cdots, r_n$ (where each of them is at most $q-1$) are said to be valid if $\sum_{i=1}^{n}(q-1-r_i) > t$.*

The rest of this section is devoted to proving Theorem 3.4.1. We will show that a weighted sum, weighed by valid exponents, over an n dimensional space can be written as a linear validly weighed sum over $(n-1)$ dimensional space as long as $(n-1) \geq d$. We begin with a simple lemma.

Lemma 3.4.4 *A function $f : \mathbb{F}_q^n \to \mathbb{F}_q$ belongs to $\mathcal{P}_t$ if and only if for every D, $(n \geq D \geq d)$ and every $y_1, \cdots, y_D, b \in \mathbb{F}_q^n$, and every valid set of exponents $\{r_i\}_{i\in[D]}$, it holds that*

$$\sum_{a=\langle a_1,\cdots,a_D\rangle \in \mathbb{F}_q^D} \prod_{i=1}^{D} a_i^{r_i} f(b + \sum_{j=1}^{D} a_j y_j) = 0. \tag{3.10}$$

Proof: If $f \in \mathcal{P}_t$ then Lemma 3.2.4 implies that for every choice of $D, y_1, \cdots, y_D, b$ and every valid choice of $r_1, \cdots, r_D$ the left hand side of Equation 3.10 evaluates to 0.

For the other direction assume that for every choice of $D, y_1, \cdots, y_D, b$ and every valid choice of $r_1, \cdots, r_D$ the function f satisfies Equation 3.10. Assume for

contradiction that $f \notin \mathcal{P}_t$. Choose $D = n$. Let m be a monomial in f with nonzero co-efficient and maximum total degree (which is at least $t+1$). Let $m(x) = \prod_{i=1}^{n} x_i^{e_i}$. For each $i \in [n]$, choose $r_i = (q - 1 - e_i)$. Clearly $\sum_{i=1}^{n}(q - 1 - r_i) = \sum_{i=1}^{n} e_i > t$. By Lemma 3.2.4 we have that

$$\sum_{a=\langle a_1,\cdots,a_n\rangle \in \mathbb{F}_q^n} \prod_{i=1}^{n} a_i^{r_i}\ m(b + \sum_{j=1}^{n} a_j y_j) \neq 0.$$

Also by the same lemma, any monomial m' that produces a non-zero sum must satisfy $m'(x) = m(x) \prod_{i \in I; \emptyset \neq I \subseteq [n]} x_i^{q-1}$. Clearly such an m' cannot exist for our choice of m. Therefore, every other monomial contributes a zero sum. This contradicts the assumption that f satisfies Equation 3.10 for every choice of $D, r_1, \cdots, r_D$. ■

Alternative Proof: Note that the evaluations of the monomials $\prod_{i=1}^{D} a_i^{r_i}$ with $\sum_{i=1}^{D}(q-1-r_i) > t$ forms $\mathrm{GRM}_q(D(q-1)-t-1, D)$ code. Since the restriction of a degree t polynomial is always a polynomial of degree at most t, the fact that $\mathrm{GRM}_q(t, D)^{\perp} = \mathrm{GRM}_q(D(q-1) - t - 1, D)$ then completes the forward direction.

The duality of $\mathrm{GRM}(t, D)$ in the special case of $D = n$ establishes the other direction. ■

Note that the above lemma requires a test for all dimension d such that $d \leq D \leq n$. However, we will prove that it is enough to restrict the test to *only* subspaces of dimension $D = d$. We achieve this the following way.

Assume we have an affine n-dimensional space, given in some basis $y_1, \cdots, y_n$ at point b. Assume that at each point $\langle a_1, \cdots, a_n\rangle$ in this affine co-ordinate system (i.e., the point has absolute co-ordinate $\sum_{i=1}^{n} a_i y_i + b$) we assign a value[10] $\prod_{i=1}^{n} a_i^{r_i}$, where $\{r_i\}_{i \in [n]}$ is a valid set of exponent. Informally lets call these geometric objects as funny-flats. Then Equation 3.9 can be interpreted as the dot product of the function with this funny-flats. We show that an n-dimension funny-flat can be written as linear combinations of $(n-1)$-dimensional funny flats provided $n - 1 \geq d$. Therefore if the dot product of f with every d-dimensional funny-flats yield zero, then for every n-dimensional funny-flats ($n \geq D$) the sum will evaluate to zero. We essentially employ this idea to prove Theorem 3.4.1.

[10]Here $r_i = 0$ can give rise to a situation 0^0. When $r_i = 0$ we simply assume that a_i term does not exist in the product, or equivalently we choose $0^0 = 1$.

We now give a definition.

Definition 3.4.5 *We define* $\delta_a(x) \stackrel{def}{=} (1-(x-a)^{q-1})$. *Note* $\delta_a(x)$ *is equal to one if and only if* $x = a$ *and zero otherwise. Also note that* $\sum_{x\in\mathbb{F}_q} \delta_a(x)g(x) = g(a)$.

Consider the following identity which holds over any finite field $\mathbb{F}_q$:

Fact 3.4.6 *If* $0 < i < q-1$, *and* $0 < j < q-1$ *and* $i + j < q-1$, *then we have*

$$z^{q-1-i}y^{q-1-j} = \sum_{v\in\mathbb{F}_q^*} \left(1-(z-vy)^{q-1}\right)\cdot v^{q-1-i}y^{q-1-i-j} = \sum_{v\in\mathbb{F}_q^*} \delta_{vy}(z)v^{q-1-i}y^{q-1-i-j} \tag{3.11}$$

Also, as a special case we have the following: for $0 < r < q$,

$$z^r = \sum_{v\in\mathbb{F}_q^*} \left(1-(z-v)^{q-1}\right)v^r = \sum_{v\in\mathbb{F}_q^*} \delta_v(z)v^r \tag{3.12}$$

If we call $q-1-i$ to be a hole corresponding to a term y^i, then Equation 3.11 essentially allows us to transfer the combined hole of y and z onto a hole of y. Also, note that $(1-(z-vy)^{q-1})$ plays a role similar to that of delta function. Thus a sum on z will vanish z all together from the right hand side. This will enable us to go down a dimension. However, note that to apply the identity, one must have the total degree in the left hand side be at least q.

Proof of Fact 3.4.6 We now establish the identity given by Equation 3.11. The identity trivially holds when $y = 0$. Moreover if $z = 0$, then $\delta_{vy}(z) = 0$ always zero unless $y = 0$ and the identity holds. Therefore assume $y, z \neq 0$. Then the only non-zero term in the right hand side corresponds to $v = z/y$. Substituting we get back the left hand side. This establishes the identity. ■

We see that in order to apply Equation 3.11, we need the total degree be at least q. The following identity will allow us to increase the degree of an individual term in units of q/p, without affecting the total hole of terms featuring a_is.

Fact 3.4.7 *For* $n \geq 2$, *for all* $i \in [n-1], 0 \leq t_i < p, \sum_{i=1}^{n-1} t_i\frac{q}{p} + r_n \leq (q-1)$, *for*

all $\langle u_1, \cdots, u_{n-1}\rangle \in \mathbb{F}_q^n$*, there exists* $c_n \in \mathbb{F}_p^* \subseteq \mathbb{F}_q^*$ *such*[11] *that*

$$\prod_{i=1}^{n-1} a_i^{t_i \frac{q}{p}} a_n^{r_n} = c_n \sum_{u=\langle u_1,\cdots,u_{n-1}\rangle \in \mathbb{F}_q^{n-1}} \prod_{i=1}^{n-1} u_i^{q-1-t_i\frac{q}{p}} \left(\sum_{j=1}^{n-1} u_j a_j + a_n\right)^{\sum_{e=1}^{n-1} t_e \frac{q}{p} + r_n} \tag{3.13}$$

Before we prove the above identity, we quickly recall the following Lemma from[12] [FS95].

Lemma 3.4.8 *([FS95], Lemma 12) Let* $0 < r \leq n \leq q-1$*. If* $r = k\frac{q}{p}$*, then* $\binom{n}{r}$ *is not divisible by* p*.*

Proof of Fact 3.4.7 We prove the identity given in Equation 3.13 by induction on n. For the base case, $n = 2$, we expand the right hand side.

$$\begin{aligned}
\sum_{u_1 \in \mathbb{F}_q} u_1^{q-1-t_1\frac{q}{p}} (u_1 a_1 + a_2)^{t_1\frac{q}{p}+r_2} &= \sum_{\substack{u_1\in\mathbb{F}_q \\ 0\leq j\leq t_1\frac{q}{p}+r_2}} \binom{t_1\frac{q}{p}+r_2}{j} u_1^{q-1-t_1\frac{q}{p}+j} a_1^j a_2^{t_1\frac{q}{p}+r_2-j} \\
&= \binom{t_1\frac{q}{p}+r_2}{t_1\frac{q}{p}} (-1) a_1^{t_1\frac{q}{p}} a_2^{r_2} \qquad \text{(Lemma 3.2.4)} \\
&= c_2^{-1} a_1^{t_1\frac{q}{p}} a_2^{r_2}
\end{aligned}$$

In the above we have used Lemma 3.2.3. Also, by Lemma 3.4.8 $c_2^{-1} \neq 0$ if $t_1 \neq 0$ and clearly $c_2 = -1$ if $t_1 = 0$.

Assume the identity holds for $n-1$. To prove it for n, we apply the identity

[11] Note c_n may depend on t_is.

[12] An alternative proof may be given using Lucas Theorem.

twice with n being $n-1$ and 2. Consider the L.H.S.

$$\begin{aligned}
\prod_{i=1}^{n-1} a_i^{t_i \frac{q}{p}} a_n^{r_n} &= a_1^{t_1 \frac{q}{p}} c_{n-1} \sum_{u=\langle u_2,\cdots,u_{n-1}\rangle \in \mathbb{F}_q^{n-2}} \prod_{i=2}^{n-1} u_i^{q-1-t_i\frac{q}{p}} (\sum_{j=2}^{n-1} u_j a_j + a_n)^{\sum_{e=2}^{n-1} t_e \frac{q}{p} + r_n} \\
&\qquad\qquad \text{(I. H.)} \\
&= c_{n-1} \sum_{u=\langle u_2,\cdots,u_{n-1}\rangle \in \mathbb{F}_q^{n-2}} \prod_{i=2}^{n-1} u_i^{q-1-t_i\frac{q}{p}} a_1^{t_1\frac{q}{p}} (\sum_{j=2}^{n-1} u_j a_j + a_n)^{\sum_{e=2}^{n-1} t_e \frac{q}{p} + r_n} \\
&= c_2 c_{n-1} \sum_{u=\langle u_1,\cdots,u_{n-1}\rangle \in \mathbb{F}_q^{n-1}} \prod_{i=1}^{n-1} u_i^{q-1-t_i\frac{q}{p}} (\sum_{j=1}^{n-1} u_j a_j + a_n)^{\sum_{e=1}^{n-1} t_e \frac{q}{p} + r_n} \\
&\qquad\qquad \text{(With } n=2\text{)} \\
&= c_n \sum_{u=\langle u_1,\cdots,u_{n-1}\rangle \in \mathbb{F}_q^{n-1}} \prod_{i=1}^{n-1} u_i^{q-1-t_i\frac{q}{p}} (\sum_{j=1}^{n-1} u_j a_j + a_n)^{\sum_{e=1}^{n-1} t_e \frac{q}{p} + r_n}
\end{aligned}$$

where $c_n = c_2 c_{n-1}$ is a new constant. This completes the induction. ■

The following Proposition will imply Theorem 3.4.1 by an easy induction on dimension D. We therefore prove the Proposition.

Proposition 3.4.9 *A function $f : \mathbb{F}_q^n \to \mathbb{F}_q$ belongs to $\mathcal{P}_t$ if and only if for every $D, (n-1 \geq D \geq d)$, every choice of $y_1, \cdots, y_D, b \in \mathbb{F}_q^n$, and every valid choice of $\{r_i\}_{i\in[D]}$, it holds that*

$$\sum_{a=\langle a_1,\cdots,a_D\rangle \in \mathbb{F}_q^D} \prod_{i=1}^{D} a_i^{r_i} \; f(b + \sum_{j=1}^{D} a_j y_j) = 0. \tag{3.14}$$

Proof: The forward direction follows from Lemma 3.4.4. We now prove the other direction. To that purpose by Lemma 3.4.4, it suffices to show the following. Assume for every choice of $y_1, \cdots, y_{n-1}, b$ and every valid choice of $r_1, \cdots, r_{n-1}$, the function f satisfies Equation 3.14. We show that for every choice of $y'_1, \cdots, y'_n, b'$ and every valid choice of $r'_1, \cdots, r'_n$, it holds that

$$\sum_{a=\langle a_1,\cdots,a_n\rangle \in \mathbb{F}_q^n} \prod_{i=1}^{n} a_i^{r'_i} f(b' + \sum_{j=1}^{n} a_j y'_j) = 0.$$

Without loss of any generality, we will assume r'_is are ordered, i.e., $0 \leq r'_1 \leq$

$\cdots \leq r'_n$. We break the proof into cases.

- **Case 1:** *There exists a subset $R \subseteq [n]$, where $|R| = n-1$ and $\sum_{i\in R}(q-1-r'_i) > t$.*
  By the imposed ordering, $R = [n-1]$ suffices. Then

$$\sum_{\langle a_1,\cdots,a_n\rangle\in\mathbb{F}_q^n} \prod_{i=1}^{n} a_i^{r'_i} f(b' + \sum_{j=1}^{n} a_j y'_j) \quad =$$

$$\sum_{a_n\in\mathbb{F}_q} a_n^{r'_n} \sum_{\langle a_1,\cdots,a_{n-1}\rangle\in\mathbb{F}_q^{n-1}} \prod_{i=1}^{n-1} a_i^{r'_i} f(b' + a_n y'_n + \sum_{j=1}^{n-1} a_j y'_j) = 0 \quad (3.15)$$

  Note that line above Equation 3.15 is a linear sum of suitably-weighted sums of f over $(n-1)$-dimensional affine subspaces, and therefore, must be zero by the assumption.

- **Case 2:** Suppose Case 1 does not hold. Then one may hope to apply Equation 3.11 to get rid of a dimension. However, note that this requires $r'_{n-1} + r'_n \geq q$. In general that may not necessarily hold[13]. However, continuing on this idea, we show how to reduce a dimension, in general. We will try to push the exponent of the term containing a_{n-1}. In that way we will either satisfy Case 1 or will be able to apply Equation 3.11. For $i \in [n-2]$, let $r'_i = s_i\frac{q}{p} + w_i$, $0 \leq w_i < \frac{q}{p}$. We claim that if case 1 cannot be applied then, $\sum_{i=1}^{n-1} s_i > 0$. For contradiction assume that $\sum_{i=1}^{n-1} s_i = 0$. Then $\sum_{i=1}^{n-1} r'_i \leq (n-1)(q/p-1)$. Hence, since case 1 does not apply, $t \geq \sum_{i=1}^{n-1}(q-1-r'_i) \geq (n-1)(q-q/p)$. Thus $d = \lceil\frac{(t+1)}{(q-q/p)}\rceil \geq n$. That contradicts the choice of $n-1$. In particular this implies that $r'_n \geq r'_{n-1} \geq \frac{q}{p}$.

  For $i = 1$ to $(n-2)$, let $0 \leq t_i \leq s_i$ be such that $\sum_{i=1}^{n-2} t_i$ is largest, and $\sum_{i=1}^{n-2} t_i \leq p-1-s_{n-1}$. Clearly such a set of $\{t_i\}_{i\in[n-1]}$ exists (not necessarily attaining the maximum). Now we show that we can push the exponent of the term containing a_{n-1} by an amount equal to $\sum_{i=1}^{n-2} t_i\frac{q}{p}$.

  To that direction we apply the identity given in Equation 3.13. The following claim proves the sufficient condition to invoke the identity.

[13]For the case $p = 2$, it follows that this always happens. We leave the details as an exercise for interested readers.

Claim 3.4.10 $\sum_{i=1}^{n-2} t_i\frac{q}{p} + r'_{n-1} \leq q - \frac{q}{p} + w_n < q.$

Proof: Since $\sum_{i=1}^{n-1} t_i \leq p-1-s_n$, we have $\sum_{i=1}^{n-1} \frac{q}{p}t_i \leq q - q/p - (r'_n - w_n)$. Since $w_{n-1} < \frac{q}{p}$, we are done. ■

Note if $\sum_{i=1}^{n-2} t_i = p-1-s_{n-1}$ holds then

$$r'_n + (r'_{n-1} + \sum_{i=1}^{n-2} t_i\frac{q}{p}) = r'_n + r'_{n-1} + (q - \frac{q}{p}) - (r'_{n-1} - w_{n-1}) = q + (r'_n - \frac{q}{p}) + w_{n-1} \geq q,$$

and therefore we will be done by applying Equation 3.11. Consequently, if $(r'_{n-1} + \sum_{i=1}^{n-2} t_i\frac{q}{p}) + r'_n < q$, then it must be the case that $\sum_{i=1}^{n-2} t_i < p-1-s_{n-1}$. This implies that for $i \in [n-2]$ $t_i = s_i$. In that case, instead of pushing $\sum_{i=1}^{n-2} t_i\frac{q}{p}$ to r'_{n-1}, we push $\sum_{i=1}^{n-1} s_i\frac{q}{p}$ to r'_n. Note in that case, $\sum_{i=1}^{n-1}(q-1-(r'_i - s_i\frac{q}{p})) > t$ must hold. For otherwise, we will have $t \geq (n-1)(q-q/p)$ that implies $d \geq n$, a contradiction. We therefore break the proof in two cases.

– **Sub-case 2.1 :** $(r'_{n-1} + \sum_{i=1}^{n-2} t_i\frac{q}{p}) + r'_n \geq q$ **holds.**

Consider $\prod_{i=1}^{n} a_i^{r'_i}$. We first rewrite this as follows

$$\prod_{i=1}^{n} a_i^{r'_i} = a_n^{r'_n} \prod_{i=1}^{n-2} a_i^{r'_i - t_i\frac{q}{p}} \times \prod_{j=1}^{n-2} \underbrace{a_j^{t_j\frac{q}{p}} \times a_{n-1}^{r'_{n-1}}}. \tag{3.16}$$

Now we apply the identity as in Equation 3.13 to the terms inside the under-brace to get

$$c_n a_n^{r'_n} \prod_{i=1}^{n-2} a_i^{r'_i - t_i\frac{q}{p}} \sum_{v \in \mathbb{F}_q^{n-2}} \left[\prod_{h=1}^{n-2} v_h^{q-1-t_h\frac{q}{p}} \cdot (\sum_{j=1}^{n-2} a_j v_j + a_{n-1})^{\sum_{e=1}^{n-2} t_e\frac{q}{p} + r'_{n-1}} \right]. \tag{3.17}$$

Note that by Claim 3.4.10, the last exponent in the above expression is less than q.

Now, if $\sum_{i=1}^{n-2} t_i\frac{q}{p} + r'_{n-1} = q-1$, we replace $(\sum_{i=1}^{n-2} a_i v_i + a_{n-1})^{\sum_{j=1}^{n-2} t_j\frac{q}{p} + r_{n-1}}$ by

$\sum_{v_{n-1} \in \mathbb{F}_q^*} \delta_{v_{n-1}}(\sum_{i=1}^{n-2} a_i v_i + a_{n-1}) = \sum_{v_{n-1} \in \mathbb{F}_q^*} \delta_{v_{n-1} - \sum_{i=1}^{n-2} a_i v_i}(a_{n-1})$, following Equation 3.12 (with r set to $q-1$). Now we consider the following sum:

$$\sum_{a_{n-1}\in\mathbb{F}_q} \delta_{v_{n-1}-\sum_{j=1}^{n-2} a_j v_j}(a_{n-1}) f(\sum_{j=1}^{n} a_j y'_j + b')$$

$$= f(\sum_{j=1}^{n-2} a_j y'_j + a_n y'_n + (v_{n-1} - \sum_{j=1}^{n-2} a_j v_j) y'_{n-1} + b')$$

$$= f(a_n y'_n + \sum_{j=1}^{n-2} a_j (y'_j - v_j y'_{n-1}) + (b' + v_{n-1} y'_{n-1})) = f(\sum_{j=1}^{n-2} a_j y''_j + a_n y''_n + b'') \tag{3.18}$$

We use Equations 3.16, 3.17 and 3.18 to write

$$\sum_{a=\langle a_1,\cdots,a_n\rangle\in\mathbb{F}_q^n} \prod_{i=1}^{n} a_i^{r'_i} f(b' + \sum_{j=1}^{n} a_j y'_j) =$$

$$\sum_{\langle v_1,\cdots,v_{n-2}\rangle\in\mathbb{F}_q^{n-2}} \sum_{v_{n-1}\in\mathbb{F}_q^*} \prod_{h=1}^{n-2} v_h^{q-1-t_h\frac{q}{p}} \quad \times$$

$$\left(\sum_{a=\langle a_1,\cdots,a_{n-2},a_n\rangle\in\mathbb{F}_q^{n-1}} a_n^{r'_n} \prod_{i=1}^{n-2} a_i^{r'_i - t_i\frac{q}{p}} f(\sum_{j=1}^{n-2} a_j y''_j + a_n y''_n + b'') \right).$$

The above equation tells that the suitably weighted sum of f over n dimension can be written as a linear sum of of suitably weighted sum of f over $(n-1)$ dimension. Since

$$\sum_{i=1}^{n-2}(q-1-r'_i+t_i\frac{q}{p}) + (q-1-r'_n) = \sum_{i=1}^{n-2}(q-1-r'_i) + \sum_{i=1}^{n-2} t_i\frac{q}{p} + (q-1-r'_n)$$

$$= \sum_{i=1}^{n-2}(q-1-r'_i) + (q-1-r'_{n-1}) + (q-1-r'_n) = \sum_{i=1}^{n}(q-1-r'_i) > t$$

each liner sum is zero by the inductive hypothesis. Therefore,

$$\sum_{a=\langle a_1,\cdots,a_n\rangle\in\mathbb{F}_q^n} \prod_{i=1}^{n} a_i^{r'_i} f(b' + \sum_{j=1}^{n} a_j y'_j) = 0.$$

Assume therefore that $\sum_{i=1}^{n-2} t_i\frac{q}{p}+r'_{n-1} < q-1$. Since $(r'_{n-1}+\sum_{i=1}^{n-2} t_i\frac{q}{p})+ r'_n \geq q$ holds, we apply Equation 3.11. Specifically we replace $a_n^{r'_n} \times (\sum_{i=1}^{n-2} a_i v_i + a_{n-1})^{\sum_{e=1}^{n-2} t_e\frac{q}{p}+r'_{n-1}}$ by

$$\sum_{v_{n-1}\in\mathbb{F}_q^*} \delta_{v_{n-1}a_n}(\sum_{i=1}^{n-2} a_i v_i + a_{n-1}) v_{n-1}^{r'_{n-1}+\sum_{e=1}^{n-2} t_e\frac{q}{p}} a_n^{(r'_n+r'_{n-1}+(\sum_{i=1}^{n-2}(t_i\frac{q}{p})))-(q-1)}.$$

We consider the following sum:

$$\begin{aligned}
&\sum_{v_{n-1}\in\mathbb{F}_q^*} v_{n-1}^{\sum_{i=1}^{n-2} t_i\frac{q}{p}+r'_{n-1}} \sum_{a_{n-1}\in\mathbb{F}_q} \delta_{v_{n-1}a_n}(\sum_{j=1}^{n-2} a_j v_j + a_{n-1}) f(\sum_{h=1}^{n} a_h y'_h + b') \quad = \\
&\sum_{v_{n-1}\in\mathbb{F}_q^*} v_{n-1}^{\sum_{i=1}^{n-2} t_i\frac{q}{p}+r'_{n-1}} \sum_{a_{n-1}\in\mathbb{F}_q} \delta_{v_{n-1}a_n-\sum_{j=1}^{n-2} a_j v_j}(a_{n-1}) f(\sum_{h=1}^{n} a_h y'_h + b') \quad = \\
&\sum_{v_{n-1}\in\mathbb{F}_q^*} v_{n-1}^{\sum_{i=1}^{n-2} t_i\frac{q}{p}+r'_{n-1}} f(\sum_{j=1}^{n-2} a_j y'_j + a_n y'_n + (v_{n-1}a_n - \sum_{h=1}^{n-2} a_h v_h) y'_{n-1} + b') \quad = \\
&\sum_{v_{n-1}\in\mathbb{F}_q^*} v_{n-1}^{\sum_{i=1}^{n-2} t_i\frac{q}{p}+r'_{n-1}} f(\sum_{j=1}^{n-2} a_j (y'_j - v_j y'_{n-1}) + a_n(y'_n + v_{n-1}y'_{n-1}) + b') \quad = \\
&\qquad\qquad \sum_{v_{n-1}\in\mathbb{F}_q^*} v_{n-1}^{\sum_{i=1}^{n-2} t_i\frac{q}{p}+r'_{n-1}} f(\sum_{j=1}^{n-2} a_j y''_j + a_n y''_n + b'')
\end{aligned} \tag{3.19}$$

We use Equations 3.16, 3.17 and 3.19 to write

$$\sum_{a=\langle a_1,\cdots,a_n\rangle\in\mathbb{F}_q^n} \prod_{i=1}^{n} a_i^{r'_i} f(b' + \sum_{j=1}^{n} a_j y'_j)$$

$$= \sum_{v=\langle v_1,\cdots,v_{n-2}\rangle\in\mathbb{F}_q^{n-2}} \sum_{v_{n-1}\in\mathbb{F}_q^*} \left[\prod_{h=1}^{n-2} v_h^{q-1-t_h\frac{q}{p}} v_{n-1}^{\sum_{e=1}^{n-2} t_e\frac{q}{p}+r'_{n-1}} \right.$$

$$\left. \left(\sum_{\substack{a_1,\cdots,a_{n-2},\\ a_n\in\mathbb{F}_q}} a_n^{r'_n+r'_{n-1}+\sum_{i=1}^{n-2} t_i\frac{q}{p}-(q-1)} \prod_{i=1}^{n-2} a_i^{r'_i-t_i\frac{q}{p}} f(\sum_{j=1}^{n-2} a_j y''_j + a_n y''_n + b'') \right) \right].$$

The above equation tells that the suitably weighted sum of f over n dimension can be written as a linear sum of of suitably weighted sum of f over $(n-1)$ dimension. Now since

$$(q-1-r'_n-r'_{n-1}-\sum_{i=2}^{n-1}t_i\frac{q}{p}+(q-1))+\sum_{i=1}^{n-2}(q-1-r'_i+t_i\frac{q}{p}) \quad =$$
$$(q-1-r'_n)+(q-1-r'_{n-1})-\sum_{i=1}^{n-2}t_i\frac{q}{p}+\sum_{i=1}^{n-2}(q-1-r'_i)+\sum_{i=1}^{n-2}t_i\frac{q}{p}$$
$$=\sum_{i=1}^{n}(q-1-r'_i)>t,$$

hence each liner sum is zero by the inductive hypothesis. Therefore,

$$\sum_{a=\langle a_1,\cdots,a_n\rangle\in\mathbb{F}_q^n}\prod_{i=1}^{n}a_i^{r'_i}f(b'+\sum_{j=1}^{n}a_jy'_j)=0.$$

– **Subcase 2.2:** $r'_n+r'_{n-1}+\sum_{i=1}^{n-2}t_i<q$ **holds.**

As mentioned previously, then it must be the case that $\sum_{i=1}^{n-2}t_i<p-1-s_{n-1}$ and hence, $t_i=s_i$ for all $i\in[n-2]$. Set $t_{n-1}=s_{n-1}$. Then $r'_n+\sum_{i=1}^{n-1}t_i\frac{q}{p}<q$. Also, note that for $i\in[n-1], r'_i-t_i\frac{q}{p}=w_i<q/p$. Denote $\rho=\sum_{i=1}^{n-1}t_i\frac{q}{p}+r'_n$. Note $0<\rho<q$.

Now consider $\prod_{i=1}^{n}a_i^{r'_i}$. We first rewrite this as follows

$$\prod_{i=1}^{n}a_i^{r'_i}=\prod_{i=1}^{n-1}a_i^{r'_i-t_i\frac{q}{p}}\times\prod_{j=1}^{n-1}\underbrace{a_j^{t_j\frac{q}{p}}\times a_n^{r'_n}}. \tag{3.20}$$

Now we apply the identity as in Equation 3.13 to the terms inside the under-brace to get

$$c_n\prod_{i=1}^{n-1}a_i^{w_i}\sum_{v=\langle v_1,\cdots,v_{n-1}\rangle\in\mathbb{F}_q^{n-1}}\left[\prod_{h=1}^{n-1}v_h^{q-1-t_h\frac{q}{p}}\cdot(\sum_{j=1}^{n-1}a_jv_j+a_n)^{\rho}\right]. \tag{3.21}$$

Since $0<\rho<q$, we replace $(\sum_{i=1}^{n-1}a_iv_i+a_n)^{\rho}$ by $\sum_{v_n\in\mathbb{F}_q^*}\delta_{v_n}(\sum_{i=1}^{n-1}a_iv_i+a_n)v_n^{\rho}=\sum_{v_n\in\mathbb{F}_q^*}\delta_{v_n-\sum_{i=1}^{n-1}a_iv_i}(a_n)v_n^{\rho}$, following Equation 3.12. Now we

consider the following sum:

$$\sum_{a_n \in \mathbb{F}_q} \delta_{v_n - \sum_{j=1}^{n-1} a_j v_j}(a_n) f(\sum_{h=1}^{n} a_h y'_h + b') = f(\sum_{h=1}^{n-1} a_h y'_h + (v_n - \sum_{j=1}^{n-1} a_j v_j) y'_n + b')$$

$$= f(\sum_{j=1}^{n-1} a_j (y'_j - v_j y'_n) + (b' + v_n y'_n)) = f(\sum_{j=1}^{n-1} a_j y''_j + b'') \qquad (3.22)$$

We use Equations 3.20, 3.21 and 3.22 to write

$$\sum_{a=\langle a_1, \cdots, a_n \rangle \in \mathbb{F}_q^n} \prod_{i=1}^{n} a_i^{r'_i} f(b' + \sum_{j=1}^{n} a_j y'_j) =$$

$$\sum_{\langle v_1, \cdots, v_{n-1} \rangle \in \mathbb{F}_q^{n-1}} \sum_{v_n \in \mathbb{F}_q^*} \prod_{s=1}^{n-1} v_s^{q-1-t_i \frac{q}{p}} v_n^{\rho} \left(\sum_{a \in \mathbb{F}_q^{n-1}} \prod_{i=1}^{n-1} a_i^{w_i} f(\sum_{j=1}^{n-1} a_j y''_j + b'') \right).$$

The above equation tells that the suitably weighted sum of f over n dimension can be written as a linear sum of of suitably weighted sum of f over $(n-1)$ dimension. However, as argued previously, it holds that $\sum_{i=1}^{n-1}(q-1-w_i) > t$. Therefore, by assumption

$$\sum_{a=\langle a_1, \cdots, a_n \rangle \in \mathbb{F}_q^n} \prod_{i=1}^{n} a_i^{r'_i} f(b' + \sum_{j=1}^{n} a_j y'_j) = 0.$$

■

3.5 A Tester for Low-degree Polynomials over Prime Fields

In this section we present and analyze a one-sided tester for $\mathcal{P}_t$. The analysis of the algorithm roughly follows the proof structure given in [RS96, AKK$^+$03]. We emphasize that the generalization from [AKK$^+$03] to our case is not straightforward. As in [RS96, AKK$^+$03] we define a self-corrected version of the (possibly corrupted) function being tested. The straightforward adoption of the analysis given in [RS96] gives reasonable bounds. However, the better bound is achieved by following the

techniques developed in [AKK+03]. In there, they show that the self-corrector function can be interpolated with overwhelming probability. However their approach appears to use special properties of $\mathbb{F}_2$ and it is not clear how to generalize their technique for arbitrary prime fields. We give a clean formulation which relies on the flats being represented through polynomials as described earlier. In particular, Claims 3.5.7, 3.5.9 and their generalization appear to require our new polynomial based view.

3.5.1 A Tester over Prime Fields

In this subsection we describe the algorithm when underlying field is $\mathbb{F}_p$.
Algorithm Test-$\mathcal{P}_t$ in $\mathbb{F}_p$

0. Let $t = (p-1) \cdot k + R$, $0 \leq R < (p-1)$. Denote $r = p - 2 - R$.
1. Uniformly and independently at random select $y_1, \cdots, y_{k+1}, b \in \mathbb{F}_p^n$.
2. If $T_f^r(y_1, \cdots, y_{k+1}, b) \neq 0$, then `reject`, else `accept`.

Theorem 3.5.1 *The algorithm* **Test-$\mathcal{P}_t$ in $\mathbb{F}_p$** *is a one-sided tester for $\mathcal{P}_t$ with a success probability at least min$(\Omega(p^{k+1}\epsilon), \frac{1}{2(k+7)p^{k+2}})$.*

Corollary 3.5.2 *Repeating the algorithm* **Test-$\mathcal{P}_t$ in $\mathbb{F}_p$** *$\Theta(\frac{1}{p^{k+1}\epsilon} + kp^k)$ times, the probability of error can be reduced to less than $1/2$.*

We will provide a general proof framework. However, for the ease of exposition we prove the main technical lemmas for the case of $\mathbb{F}_3$. The proof idea in the general case is similar and the details are omitted. Therefore we will essentially prove the following.

Theorem 3.5.3 *The algorithm* **Test-$\mathcal{P}_t$ in $\mathbb{F}_3$** *is a one-sided tester for $\mathcal{P}_t$ with success probability at least min$(\Omega(3^{k+1}\epsilon), \frac{1}{2(t+7)3^{t/2+1}})$.*

3.5.2 Analysis of Algorithm *Test-$\mathcal{P}_t$*

In this subsection we analyze the algorithm described in Section 3.5.1. From Claim 3.3.1 it is clear that if $f \in \mathcal{P}_t$, then the tester accepts. Thus, the bulk of the proof is to show that if f is ϵ-`far` from $\mathcal{P}_t$, then the tester rejects with significant probability. Our proof structure follows that of the analysis of the test in [AKK+03]. In what

follows, we will denote $T_f(y_1, \cdots, y_l, b)$ by $T_f^0(y_1, \cdots, y_l, b)$ for the ease of exposition. In particular, let f be the function to be tested for membership in $\mathcal{P}_t$. Assume we perform Test T_f^i for an appropriate i as required by the algorithm described in Section 3.5.1. For such an i, we define $g_i : \mathbb{F}_p^n \to \mathbb{F}_p$ as follows: For $y \in F_p^n, \alpha \in \mathbb{F}_p$, denote $p_{y,\alpha} = \Pr_{y_1,\cdots,y_{k+1}}[f(y) - T_{f_i}(y - y_1, y_2, \cdots, y_{k+1}, y_1) = \alpha]$. Define $g_i(y) = \alpha$ such that $\forall \beta \neq \alpha \in \mathbb{F}_p, p_{y,\alpha} \geq p_{y,\beta}$ with ties broken arbitrarily. With this meaning of plurality, for all $i \in [p-2] \cup \{0\}$, g_i can be written as:

$$g_i(y) \quad = \quad \text{plurality}_{y_1,\cdots,y_{k+1}} \left[f(y) - T_f^i(y - y_1, y_2, \cdots, y_{k+1}, y_1) \right]. \tag{3.23}$$

Further we define

$$\eta_i \stackrel{def}{=} Pr_{y_1,\cdots,y_{k+1},b}[T_f^i(y_1, \cdots, y_{k+1}, b) \neq 0] \tag{3.24}$$

The next lemma follows from a Markov-type argument.

Lemma 3.5.4 *For a fixed $f : \mathbb{F}_p^n \to \mathbb{F}_p$, let g_i, η_i be defined as above. Then, $\delta(f, g_i) \leq 2\eta_i$.*

Proof: Consider the set of elements y such that $\Pr_{y_1,\cdots,y_{k+1}}[f(y) = f(y) - T_f^i(y - y_1, y_2, \cdots, y_{k+1}, y_1)] < 1/2$. If the fraction of such elements is more than $2\eta_i$ then that contradicts the condition that

$$\begin{aligned} \eta_i &= \Pr_{y_1,\cdots,y_{k+1},b}[T_f^i(y_1, \cdots, y_{k+1}, b) \neq 0] \\ &= \Pr_{y_1,y_2,\cdots,y_{k+1},b}[T_f^i(y_1 - b, y_2, \cdots, y_{k+1}, b) \neq 0] \\ &= \Pr_{y,y_1,\cdots,y_{k+1}}[f(y) \neq f(y) - T_f^i(y - y_1, y_2, \cdots, y_{k+1}, y_1)]. \end{aligned}$$

Therefore, we obtain $\delta(f, g_i) \leq 2\eta_i$. ■

Note that $\Pr_{y_1,\cdots,y_{k+1}}[g_i(y) = f(y) - T_f^i(y - y_1, y_2, \cdots, y_{k+1}, y_1)] \geq \frac{1}{p}$. We now show that this probability is actually much higher. The next lemma gives a weak bound in that direction following the analysis in [RS96]. For the sake of completeness, we present a proof in the Appendix A.1.

Lemma 3.5.5 $\forall y \in \mathbb{F}_p^n$, $\Pr_{y_1,\cdots,y_{k+1} \in \mathbb{F}_p^n}[g_i(y) = f(y) - T_f^i(y - y_1, y_2, \cdots, y_{k+1}, y_1)] \geq 1 - 2p^{k+1}\eta_i$.

However, when the degree being tested is larger than the field size, we can improve the above lemma considerably. The following lemma strengthens Lemma 3.5.5 when $t \geq (p-1)$ or equivalently $k \geq 1$. We now focus on the $\mathbb{F}_3$ case.

Lemma 3.5.6 $\forall y \in \mathbb{F}_3^n$, $\Pr_{y_1,\cdots,y_{k+1}\in\mathbb{F}_3^n}[g_i(y) = f(y) - T_f^i(y - y_1, y_2, \cdots, y_{k+1}, y_1)] \geq 1 - (4k+14)\eta_i$.

Observe that the goal of the lemma is to show that at any fixed point y, if g_i is interpolated out of a random hyperplane, then w.h.p. the interpolated value is the most popular vote. To ensure this we show that if g_i is interpolated on two independently random hyperplanes, then the probability that these interpolated values are same, that is the collision probability, is large. To estimate this collision probability, we show that the difference of the interpolation values can be rewritten as a sum of T_f^i on small number of random hyperplanes. Thus if the test passes often (that is, T_f^i evaluates to zero w.h.p.), then this sum (by a simple union bound) evaluates to zero often, which proves the high collision probability.

The improvement will arise because we will express differences involving $T_f^i(\cdots)$ as a telescoping series to essentially reduce the number of events in the union bound. To do this we will need the following claims. They can easily be verified by expanding the terms on both sides like the proof of Claim 4 in [AKK$^+$03]. However, this does not give much insight into the general case, i.e., for $\mathbb{F}_p$. We provide an alternate proof that can be generalized to get similar claims and has a much cleaner structure based on the underlying geometric structure, i.e., flats or pseudoflats.

Claim 3.5.7 *For every $l \in \{2, \cdots, k+1\}$, for every $y(= y_1), z, w, b, y_2, \cdots, y_{l-1}$, $y_{l+1}, \cdots, y_{k+1} \in \mathbb{F}_3^n$, let*

$$S_f(y, z) \stackrel{def}{=} T_f(y, y_2, \cdots, y_{l-1}, z, y_{l+1}, \cdots, y_{k+1}, b).$$

Then[14] *the following holds:*

$$S_f(y, w) - S_f(y, z) = S_f(y + w, z) + S_f(y - w, z) - S_f(y + z, w) - S_f(y - z, w).$$

[14] Note that $T_f(\cdot)$ is a symmetric function in all but its last input. Therefore to enhance readability, we omit the reference to index i.

Proof: Assume y, z, w are independent[15]. We claim that it is enough to prove the result for $k = 1$ and $b = \mathbf{0}$. A linear transform (or renaming the co-ordinate system appropriately) reduces the case of $k = 1$ and $b \neq \mathbf{0}$ to the case of $k = 1$ and $b = \mathbf{0}$. We now show how to "reduce" the case of $k > 1$ to the $k = 1$ case. Fix some values $c_2, \cdots, c_{l-1}, c_{l+1}, \cdots, c_{k+1}$ and note that one can write $c_1 y + c_2 y_2 + \cdots c_{l-1} y_{l-1} + c_l w + c_{l+1} y_{l+1} + c_{k+1} y_{k+1} + b$ as $c_1 y + c_l w + b'$, where $b' = \sum_{j \in \{2,\cdots,l-1,l+1,\cdots,k+1\}} c_j y_j + b$. Thus, $S_f(y, w) = \sum_{(c_2,\cdots,c_{l-1},c_{l+1},\cdots,c_{k+1}) \in \mathbb{F}_3}^{k-1} \sum_{(c_1,c_l) \in \mathbb{F}_3^2} f(c_1 y + c_l w + b')$, where b' is as defined earlier. One can rewrite the other $S_f(\cdot)$ terms similarly. Note that for a fixed vector $(c_2, \cdots, c_{l-1}, c_{l+1}, \cdots, c_{k+1})$, the value of b' is the same. Finally note that the equality (in the $k > 1$ case) is satisfied if 3^{k-1} similar equalities hold (in the $k = 1$ case).

Now consider the space $\mathcal{H}$ generated by y, z and w at $\mathbf{0}$. Note that $S_f(y, w)$ (with $b = \mathbf{0}$) is just $f \cdot 1_L$, where 1_L is the incidence vector of the flat given by the equation $z = 0$. Therefore 1_L is equivalent to the polynomial $(1 - z^2)$. Similarly $S_f(y, z) = f \cdot 1_{L'}$ where L' is given by the equation $(1 - w^2)$. We use the following polynomial identity (in $\mathbb{F}_3$)

$$w^2 - z^2 = [1 - (y - w)^2 + 1 - (y + w)^2] - [1 - (y + z)^2 + 1 - (y - z)^2].$$

Now observe that the equation $(1 - (y - w)^2)$ is the incidence vector of the flat generated by $y + w$ and z. Similar observations hold for other terms. Therefore, interpreting the above equation in terms of incidence vectors of flats, we complete the proof with Observation 3.3.5. ■

We have the following analogue[16] of Claim 3.5.7 in $\mathbb{F}_p$:

Claim 3.5.8 *For every $l \in \{2, \cdots, k+1\}$, for every $y(= y_1), z, w, b, y_2, \cdots, y_{l-1}$,*

[15] If not then both sides are equal to 0 and hence the equality is trivially satisfied. To see why this claim is true for the left hand side, recall the definition of $T_f(\cdot)$ and note that the sets of points in the flat generated by $y, y_2, \cdots, y_{l-1}, w, y_{l+1}, \cdots, y_{k+1}$ at b and the flat generated by $y, y_2, \cdots, y_{l-1}, z, y_{l+1}, \cdots, y_{k+1}$ at b are the same. A similar argument works for the expression on the right hand side of the equality.

[16] This claim can be extended to $\mathbb{F}_q$ in a straightforward manner. We mention here that this lemma over $\mathbb{F}_q$ allows one to prove a similar version of Lemma 3.5.6 over $\mathbb{F}_q$. That lemma along with versions of Lemma 3.5.15 and Lemma 3.5.20 can be used to get a robust characterization as is done in [KR04].

$y_{l+1}, \cdots, y_{k+1} \in \mathbb{F}_p^n$, *with notation used from the previous lemma, it holds that*

$$S_f(y,w) - S_f(y,z) = \sum_{e \in \mathbb{F}_p^*} \left[S_f(y+ew,z) - S_f(y+ez,w) \right].$$

Proof: (**Sketch**) Consider the following identity

$$w^{(p-1)} - z^{(p-1)} = \sum_{e \in \mathbb{F}_p^*} \Big[[1-(ew+y)^{(p-1)}] - [1-(ez+y)^{(p-1)}] \Big] \tag{3.25}$$

then we can prove the claim along the same lines as the alternate proof of Claim 3.5.7. We complete the proof by proving Equation 3.25. Consider the sum: $\sum_{e \in \mathbb{F}_p^*} (ew+y)^{(p-1)}$. Expanding the terms and rearranging the sums we get

$$\sum_{j=0}^{(p-1)} \binom{(p-1)}{j} w^{(p-1)-j} y^j \sum_{e \in \mathbb{F}_p^*} e^{p-1-j}.$$

By Lemma 3.2.3 the sum evaluates to $(-w^{(p-1)} - y^{(p-1)})$. Similarly, $\sum_{e \in \mathbb{F}_p^*} (ez+y)^{(p-1)} = (-z^{(p-1)} - y^{(p-1)})$ which proves Equation 3.25. ■

Claim 3.5.9 *For every* $l \in \{2, \cdots, k+1\}$, *for every* $y(=y_1), z, w, b, y_2, \cdots, y_{l-1}, y_{l+1}, \cdots, y_{k+1} \in \mathbb{F}_3^n$, *denote*

$$S_f^1(y,w) \stackrel{def}{=} T_f^1(y, y_2, \cdots, y_{l-1}, w, y_{l+1}, \cdots, y_{k+1}, b).$$

Then[17] *the following holds:*

$$S_f^1(y,w) - S_f^1(y,z) = S_f^1(y+z,w) + S_f^1(y-z,w) - S_f^1(y+w,z) - S_f^1(y-w,z).$$

Proof: Note here that the defining equation of $S_f^1(y,z)$ is $y(1-w^2)$. Now consider the following identity in $\mathbb{F}_3$:

$$\begin{aligned} y(z^2-w^2) &= (y+w)[1-(y-w)^2] + (y-w)[1-(y+w)^2] \\ &\quad -(y+z)[1-(y-z)^2] - (y-z)[1-(y+z)^2 \end{aligned}$$

[17] Note that $T_f^i(\cdot)$ is a symmetric function in its all but last and first input. Therefore to enhance readability, we omit the reference to index l.

for variables $y, z, w \in \mathbb{F}_3$. Rest of the proof is similar to the proof of Claim 3.5.7 (the proof replaces flats by pseudoflats) and is omitted. ■

We now prove the following analogue in $\mathbb{F}_p$:

Claim 3.5.10 *For every $i \in \{1, \cdots, p-2\}$, for every $l \in \{2, \cdots, k+1\}$ and for every*
$y(= y_1), z, w, b, y_2, \cdots, y_{l-1}, y_{l+1}, \cdots, y_{k+1} \in \mathbb{F}_p^n$, denote

$$S_f^i(y, w) \stackrel{def}{=} T_f^i(y, y_2, \cdots, y_{l-1}, w, y_{l+1}, \cdots, y_{k+1}, b).$$

Then there exists c_i such that

$$S_f^i(y, w) - S_f^i(y, z) = c_i \sum_{e \in \mathbb{F}_p^*} \left[S_f^i(y + ew, z) - S_f^i(y + ez, w) \right].$$

Proof: Observe that $T_f^i(y, z) = f \cdot E_{L_i}$, where E_{L_i} denotes the evaluation vector of the pseudoflat L with exponent i, generated by y, z at b exponentiated along y. Note that the polynomial defining E_{L_i} is just $y^i(w^{(p-1)} - 1)$. We now give an identity similar to that of Equation 3.25 that completes the proof. We claim that the following identity holds

$$y^i(w^{p-1} - z^{p-1}) = c_i \sum_{e \in \mathbb{F}_p^*} \left[(y + ew)^i [1 - (y - ew)^{p-1}] - (y + ez)^i [1 - (y - ez)^{p-1}] \right]. \tag{3.26}$$

where $c_i = 2^i$. Before we prove the identity, note that $(-1)^j \binom{p-1}{j} = 1$ in $\mathbb{F}_p$. This is because for $1 \leq m \leq j$, $m = (-1)(p - m)$. Therefore $j! = (-1)^j \frac{(p-1)!}{(p-j-1)!}$ holds in $\mathbb{F}_p$. Substitution yields the desired result. Also note that $\sum_{e \in \mathbb{F}_p^*} (y + ew)^i = -y^i$

(expand and apply Lemma 3.2.3). Now consider the sum

$$\begin{aligned}
&\sum_{e\in\mathbb{F}_p^*}(y+ew)^i(y-ew)^{(p-1)} \quad = \\
&\sum_{e\in\mathbb{F}_p^*}\sum_{\substack{0\le j\le i\\ 0\le m\le(p-1)}}(-1)^m\binom{i}{j}\binom{p-1}{m}y^{(p-1)+i-j-m}w^{j+m}e^{j+m} \\
&= \sum_{\substack{0\le j\le i\\ 0\le m\le(p-1)}}(-1)^m\binom{i}{j}\binom{p-1}{m}y^{(p-1)+i-j-m}w^{j+m}\sum_{e\in\mathbb{F}_p^*}e^{j+m} \\
&= (-1)[y^{(p-1)+i}+(-1)^{(p-1)}\sum_{j=0}^{i}\binom{i}{j}\underbrace{\binom{p-1}{p-1-j}(-1)^j}_{=1}y^iw^{(p-1)}] \\
&\qquad = \quad (-1)[y^i+y^iw^{(p-1)}2^i] \qquad (3.27)
\end{aligned}$$

Similarly one has $\sum_{e\in\mathbb{F}_p^*}(y+ez)^i(y-ez)^{(p-1)}=(-1)[y^i+y^iz^{(p-1)}2^i]$. Substituting and simplifying one gets Equation 3.26. ■

We will also need the following claims.

Claim 3.5.11 *For every $l\in\{2,\cdots,k+1\}$, $y(=y_l),z,w,b,y_2,\cdots,y_{l-1}$, $y_{l+1},\cdots,y_{k+1}\in\mathbb{F}_3^n$, with notation used in previous claim, it holds that*

$$S_f^1(w,y)-S_f^1(z,y)=S_f^1(z+w,y-z)-S_f^1(z+w,y-w)+S_f^1(y+z,w)+S_f^1(y-z,w)$$
$$-S_f^1(y+w,z)-S_f^1(y-w,z).$$

Proof: The above follows from the identity

$$w(1-z^2)-z(1-w^2)=(z+w)[1-(z+y)^2-1+(y+w)^2]+y(w^2-z^2)$$

Also we can expand $y(w^2-z^2)$ as in the proof of Claim 3.5.9. ■

We have the following analogue in $\mathbb{F}_p$.

Claim 3.5.12 *For every $i\in\{1,\cdots,p-2\}$, for every $l\in\{2,\cdots,l+1\}$ and for every*

$y(=y_l), z, w, b, y_2, \cdots, y_{l-1}, y_{l+1}, \cdots, y_{k+1} \in \mathbb{F}_p^n$, *there exists* $c_i \in \mathbb{F}_p^*$ *such that*

$$\begin{aligned} S_f^i(w,y) - S_f^i(z,y) &= \sum_{e \in \mathbb{F}_p^*} \big[S_f^i(y+ew, y-ew) - S_f^i(w+ey, w-ey) \\ &+ S_f^i(z+ey, z-ey) - S_f^i(y+ez, y-ez) + c_i \left[S_f^i(y+ew, z) - S_f^i(y+ez, w) \right] \big] \end{aligned}$$

Proof: The above follows from the identity

$$w^i(1-z^{p-1}) - z^i(1-w^{p-1}) = (w^i - y^i)(1-z^{p-1}) - (z^i - y^i)(1-w^{p-1}) + y^i(w^{p-1} - z^{p-1}).$$

We also use that $\sum_{e \in \mathbb{F}_p^*} (w+ey)^i = -w^i$ and Claim 3.5.10 to expand the last term. Note that $c_i = 2^i$ as before. ■

Proof of Lemma 3.5.6: We first prove the lemma for $g_0(y)$. We fix $y \in \mathbb{F}_3^n$ and let $\gamma \stackrel{def}{=} \Pr_{y_1, \cdots, y_{k+1} \in \mathbb{F}_3^n}[g_0(y) = f(y) - T_f(y - y_1, y_2, \cdots, y_{k+1}, y_1)]$. Recall that we want to lower bound γ by $1 - (4k+14)\eta_0$. In that direction, we bound a slightly different but related probability. Let

$$\mu \stackrel{def}{=} \Pr_{\substack{y_1, \cdots, y_{k+1}, \\ z_1, \cdots, z_{k+1} \in \mathbb{F}_3^n}} [T_f(y - y_1, y_2, \cdots, y_{k+1}, y_1) = T_f(y - z_1, z_2, \cdots, z_{k+1}, z_1)]$$

Denote $Y = \langle y_1, \cdots, y_{k+1} \rangle$ and similarly Z. Then by the definitions[18] of μ and γ we have, $\gamma \geq \mu$.

We have $\mu = \Pr_{y_1, \cdots, y_{k+1}, z_1, \cdots, z_{k+1} \in \mathbb{F}_3^n}[T_f(y - y_1, y_2, \cdots, y_{k+1}, y_1) - T_f(y - z_1, z_2, \cdots, z_{k+1}, z_1) = 0]$.

Now, for any choice of $y_1, \cdots, y_{k+1}$ and $z_1, \cdots, y_{k+1}$:

[18] Note for a probability vector $v \in [0,1]^n$, $\left\| v \right\|_\infty = \mathrm{Max}_{i \in [n]}\{v_i\} \geq \mathrm{Max}_{i \in [n]}\{v_i\} \cdot (\sum_{i=1}^n v_i) = \sum_{i=1}^n v_i \cdot \mathrm{Max}_{i \in [n]}\{v_i\} \geq \sum_{i=1}^n v_i^2 = \left\| v \right\|_2^2$.

$$
\begin{array}{lcll}
T_f(y-y_1,y_2,\cdots,y_{k+1},y_1) & - & T_f(y-z_1,z_2,\cdots,z_{k+1},z_1) & = \\
T_f(y-y_1,y_2,\cdots,y_{k+1},y_1) & - & T_f(y-y_1,y_2,\cdots,y_k,z_{k+1},y_1) & + \\
T_f(y-y_1,y_2,\cdots,y_k,z_{k+1},y_1) & - & T_f(y-y_1,y_2,\cdots,y_{k-1},z_k,z_{k+1},y_1) & + \\
T_f(y-y_1,y_2,\cdots,y_{k-1},z_k,z_{k+1},y_1) & - & T_f(y-y_1,y_2,\cdots,y_{k-2},z_{k-1},z_k,z_{k+1},y_1) & + \\
\vdots & & & \\
T_f(y-y_1,z_2,z_3,\cdots,z_{k+1},y_1) & - & T_f(y-z_1,z_2,\cdots,z_{k+1},y_1) & + \\
T_f(y-z_1,z_2,z_3,\cdots,z_{k+1},y_1) & - & T_f(y-y_1,z_2,\cdots,z_{k+1},z_1) & + \\
T_f(y-y_1,z_2,z_3,\cdots,z_{k+1},z_1) & - & T_f(y-z_1,z_2,\cdots,z_{k+1},z_1) &
\end{array}
$$

Consider any pair

$$T_f(y-y_1,y_2,\cdots,y_l,z_{l+1},\cdots,z_{k+1},y_1)-T_f(y-y_1,y_2,\cdots,y_{l-1},z_l,\cdots,z_{k+1},y_1)$$

that appears in the first k "rows" in the sum above. Note that $T_f(y-y_1,y_2,\cdots,y_l,z_{l+1},\cdots,z_{k+1},y_1)$ and $T_f(y-y_1,y_2,\cdots,y_{l-1},z_l,\cdots,z_{k+1},y_1)$ differ only in a single parameter. We apply Claim 3.5.7 and obtain:

$$
\begin{aligned}
&T_f(y-y_1,y_2,\cdots,y_l,z_{l+1},\cdots,z_{k+1},y_1)-T_f(y-y_1,y_2,\cdots,y_{l-1},z_l,\cdots,z_{k+1},y_1)= \\
&T_f(y-y_1+y_l,y_2,\cdots,y_{l-1},z_l,\cdots,z_{k+1},y_1)+T_f(y-y_1-y_l,y_2,\cdots,y_{l-1},z_l,\cdots,z_{k+1},y_1) \\
&-T_f(y-y_1+z_l,y_2,\cdots,y_l,z_{l+1},\cdots,z_{k+1},y_1)-T_f(y-y_l-z_l,y_2,\cdots,y_l,z_{l+1},\cdots,z_{k+1},y_1).
\end{aligned}
$$

Recall that y is fixed and $y_2,\cdots,y_{k+1},z_2,\cdots,z_{k+1}\in\mathbb{F}_3^n$ are chosen uniformly at random, so all the parameters on the right hand side of the equation are independent and uniformly distributed. Similarly one can expand the pairs $T_f(y-y_1,z_2,z_3,\cdots,z_{k+1},y_1)-T_f(y-z_1,z_2,\cdots,z_{k+1},y_1)$ and $T_f(y-y_1,z_2,z_3,\cdots,z_{k+1},z_1)-T_f(y-z_1,z_2,\cdots,z_{k+1},z_1)$ into four T_f with all parameters being independent and uniformly distributed[19]. Finally notice that the parameters in both $T_f(y-z_1,z_2,z_3,\cdots,z_{k+1},y_1)$ and $T_f(y-z_1,z_2,\cdots,z_{k+1},y_1)$ are independent and uniformly distributed. Further recall that by the definition of η_0, $\Pr_{r_1,\cdots,r_{k+1}}[T_f(r_1,\cdots,r_{k+1})\neq 0]\leq\eta_0$ for independent and uniformly distributed

[19]Since $T_f(\cdot)$ is symmetric.

r_is. Thus, by the union bound, we have:

$$\Pr_{\substack{y_1,\cdots,y_{k+1},\\ z_1,\cdots,z_{k+1}\in\mathbb{F}_3^n}}[T_f(y_1,\cdots,y_{k+1})-T_f(z_1,\cdots,z_{k+1})\neq 0]\leq(4k+10)\eta_0\leq(4k+14)\eta_0. \tag{3.28}$$

Therefore $\gamma\geq\mu\geq 1-(4k+14)\eta_0$. A similar argument[20] proves the Lemma for $g_1(y)$. ■

Remark 3.5.13 *Analogously, in the case $\mathbb{F}_p$ we have: for every $y\in\mathbb{F}_p^n$,* $\Pr_{y_1,y_2,\cdots,y_{k+1}\in\mathbb{F}_p^n}[g_i(y)=f(y)-T_f^i(y-y_1,y_2,\cdots,y_{k+1},y_1)+f(y)]\geq 1-2((p-1)k+6(p-1)+1)\eta_i$.
The proof is similar to that of Lemma 3.5.6 where it can be shown $\mu_i\geq 1-2((p-1)k+6(p-1)+1)\eta_i$, for each μ_i defined for $g_i(y)$.

Remark 3.5.14 *Using Lemma 3.5.6, we can get a slightly stronger version of Lemma 3.5.4 following the proof of Lemma 2 in [AKK+03]. For a fixed function $f:\mathbb{F}_p^n\to\mathbb{F}_p$, let g_i,η_i be defined as in Equations 3.23 and 3.24. Then, $\delta(f,g_i)\leq\ min(2\eta_i,\frac{\eta_i}{1-2((p-1)k+6(p-1)+1)\eta_i})$.*

The next lemma shows that sufficiently small η_i implies that g_i is the self-corrected version of the function f.

Lemma 3.5.15 *Over $\mathbb{F}_3$, if $\eta_i<\frac{1}{2(2k+7)3^{k+1}}$, then the function g_i belongs to $\mathcal{P}_t$ (assuming $k\geq 1$).*

Proof : From Theorem 3.3.1, it suffices to prove that if $\eta_i<\frac{1}{2(2k+7)3^{k+1}}$ then $T_{g_i}^i(y_1,\cdots,y_{k+1},b)=0$ for every $y_1,\cdots,y_{k+1},b\in\mathbb{F}_3^n$. Fix the choice of $y_1,\cdots,y_{k+1},b$. Define $Y=\langle y_1,\cdots,y_{k+1}\rangle$. We will express $T_{g_i}^i(Y,b)$ as the sum of $T_f^i(\cdot)$ with random arguments. We uniformly select $(k+1)^2$ random variables $z_{i,j}$ over $\mathbb{F}_3^n$ for $1\leq i\leq k+1$, and $1\leq j\leq k+1$. Define $Z_i=\langle z_{i,1},\cdots,z_{i,k+1}\rangle$. We also select uniformly $(k+1)$ random variables r_i over $\mathbb{F}_3^n$ for $1\leq i\leq k+1$. We use $z_{i,j}$ and r_i's to set up the random arguments. Now by Lemma 3.5.6, for every $I\in\mathbb{F}_3^{k+1}$ (i.e. think of I as an ordered $(k+1)$-tuple over $\{0,1,2\}$), with probability at least

[20] $T_{f_1}(.)$ is not symmetric and needs some work. We use another identity as given in Lemma 3.5.11 to resolve the issue and get four extra terms than in the case of g_0. The proof for $g_1(y)$ is same as the proof for $g_0(y)$ except it also needs Lemma 3.5.11.

$1-2(2k+7)\eta_i$ over the choice of $z_{i,j}$ and r_i,

$$g_i(I\cdot Y+b) = f(I\cdot Y+b) - T_f^i(I\cdot Y+b-I\cdot Z_1-r_1, I\cdot Z_2+r_2, \cdots, I\cdot Z_{k+1}+r_{k+1}, I\cdot Z_1+r_1), \tag{3.29}$$

where for vectors $X, Y \in \mathbb{F}_3^{k+1}$, $Y \cdot X = \sum_{i=1}^{k+1} Y_i X_i$, holds.

Let E_1 be the event that Equation 3.29 holds for all $I \in \mathbb{F}_3^{k+1}$. By the union bound:

$$\Pr[E_1] \geq 1 - 3^{k+1} \cdot 2(2k+7)\eta_i. \tag{3.30}$$

Assume that E_1 holds. We now need the following claims. Let $J = \langle J_1, \cdots, J_{k+1} \rangle$ be a $(k+1)$ dimensional vector over $\mathbb{F}_3$, and denote $J' = \langle J_2, \cdots, J_{k+1} \rangle$.

Claim 3.5.16 *If Equation 3.29 holds for all $I \in \mathbb{F}_3^{k+1}$, then*

$$\begin{aligned} T_{g_0}^0(Y,b) &= \sum_{0 \neq J' \in \mathbb{F}_3^k} \left[-T_f(y_1 + \sum_{t=2}^{k+1} J_t z_{t,1}, \cdots, y_{k+1} + \sum_{t=2}^{k+1} J_t z_{t,(k+1)}, b + \sum_{t=2}^{k+1} J_t r_t) \right] \\ &+ \sum_{J' \in \mathbb{F}_3^k} \Bigg[-T_f(2y_1 - z_{1,1} + \sum_{t=2}^{k+1} J_t z_{t,1}, \cdots, 2y_{k+1} - z_{1,(k+1)} + \sum_{t=2}^{k+1} J_t z_{t,(k+1)}, \\ &\quad 2b - r_1 + \sum_{t=2}^{k+1} J_t r_t) + T_f(z_{1,1} + \sum_{t=2}^{k+1} J_t z_{t,1}, \cdots, z_{1,k+1} + \sum_{t=2}^{k+1} J_t z_{t,(k+1)}, \\ &\quad r_1 + \sum_{t=2}^{k+1} J_t r_t) \Bigg] \end{aligned} \tag{3.31}$$

Claim 3.5.17 *If Equation 3.29 holds for all $I \in \mathbb{F}_3^{k+1}$, then*

$$\begin{aligned} T_{g_1}^1(Y,b) &= \sum_{0 \neq J' \in \mathbb{F}_3^k} \left[-T_f^1(y_1 + \sum_{t=2}^{k+1} J_t z_{t,1}, \cdots, y_{k+1} + \sum_{t=2}^{k+1} J_t z_{t,(k+1)}, b + \sum_{t=2}^{k+1} J_t r_t) \right] \\ &+ \sum_{J' \in \mathbb{F}_3^k} \Bigg[T_f^1(2y_1 - z_{1,1} + \sum_{t=2}^{k+1} J_t z_{t,1}, \cdots, 2y_{k+1} - z_{1,(k+1)} + \sum_{t=2}^{k+1} J_t z_{t,(k+1)}, \\ &\quad 2b - r_1 + \sum_{t=2}^{k+1} J_t r_t) \Bigg]. \end{aligned} \tag{3.32}$$

To maintain the flow of the proof, the proofs of Claim 3.5.16 and Claim 3.5.17 are deferred to the appendix A.1. Let E_2 be the event that for every $J' \in \mathbb{F}_3^k, T_f^i(y_1 + \sum_t J_t z_{t,1}, \cdots, y_{k+1} + \sum_t J_t z_{t,(k+1)}, b + \sum_{t=2} k+1 J_t r_t) = 0$, $T_f^i(2y_1 - z_{1,1} + \sum_{t=2}^{k+1} J_t z_{t,1}, \cdots, 2y_{k+1} - z_{1,k+1} + \sum_{t=2}^{k+1} J_t z_{t,(k+1)}, 2b - r_1 + \sum_{t=2}^{k+1} J_t r_t) = 0$, and $T_f(z_{1,1} + \sum_{t=2}^{k+1} J_t z_{t,1}, \cdots, z_{1,k+1} + \sum_{t=2}^{k+1} J_t z_{t,k+1}, r_1 + \sum_{t=2}^{k+1} J_t r_t) = 0$. By the definition of η_i and the union bound, we have:

$$\Pr[E_2] \geq 1 - 3^{k+1}\eta_i. \tag{3.33}$$

Suppose that $\eta_i \leq \frac{1}{2(2k+7)3^{k+1}}$ holds. Then by Equations 3.30 and 3.33, the probability that E_1 and E_2 hold is strictly positive. In other words, there exists a choice of the $z_{i,j}$'s and r_i's for which all summands in either Claim 3.5.16 or in Claim 3.5.17, whichever is appropriate, is 0. This implies that $T_{g_i}^i(y_1, \cdots, y_{k+1}, b) = 0$. In other words, if $\eta_i \leq \frac{1}{2(2k+7)3^{k+1}}$, then g_i belongs to $\mathcal{P}_t$. ■

Remark 3.5.18 *Over $\mathbb{F}_p$ we have: if $\eta_i < \frac{1}{2((p-1)k+6(p-1)+1)p^{k+1}}$, then g_i belongs to $\mathcal{P}_t$ (if $k \geq 1$).*

In case of $\mathbb{F}_p$, we can generalize Equation 3.29 in a straightforward manner. Let E_1' denote the event that all such events holds. We can similarly obtain

$$\Pr[E_1'] \geq 1 - p^{k+1} \cdot 2((p-1)k + 6(p-1) + 1)\eta_i. \tag{3.34}$$

Claim 3.5.19 *Assume equivalent of Equation 3.29 holds for all $I \in \mathbb{F}_p^{k+1}$, then*[21]

$$\begin{aligned} T^i_{g_i}(Y,b) &= \sum_{0\neq J'\in\mathbb{F}_p^k}\left[-T^i_f(y_1+\sum_{t=2}^{k+1}J_t z_{t,1},\cdots,y_{k+1}+\sum_{t=2}^{k+1}J_t z_{t,(k+1)},b+\sum_{t=2}^{k+1}J_t r_t)\right] \\ &+ \sum_{J'\in\mathbb{F}_p^k}\left[\sum_{J_1\in\mathbb{F}_p;J_1\neq 1}J_1^i\left[-T^i_f(J_1y_1-(J_1-1)z_{1,1}+\sum_{t=2}^{k+1}J_t z_{t,1},\cdots,\right.\right. \\ &\quad J_1y_{k+1}-(J_1-1)z_{1,(k+1)} \\ &\quad \left.\left.+\sum_{t=2}^{k+1}J_t z_{t,(k+1)},J_1b-(J_1-1)r_1+\sum_{t=2}^{k+1}J_t r_t)\right]\right] \end{aligned} \tag{3.35}$$

Let E_2' be the event analogous to the event E_2 in Claim 3.5.17. Then by the definition of η_i and the union bound, we have

$$\Pr[E_2'] \geq 1 - 2p^{k+1}\eta_i. \tag{3.36}$$

Then if we are given that $\eta_i < \frac{1}{2((p-1)k+6(p-1)+1)p^{k+1}}$, then the probability that E_1' and E_2' hold is strictly positive. Therefore, this implies $T^i_{g_i}(y_1,\cdots,y_{k+1},b)=0$.

By combining Lemma 3.5.4 and Lemma 3.5.15 we obtain that if f is $\Omega(1/(k3^k))$-far from $\mathcal{P}_t$ then $\eta_i = \Omega(1/(k3^k))$. We next consider the case in which η_i is small. By Lemma 3.5.4, in this case, the distance $\delta = \delta(f,g)$ is small. The next lemma shows that in this case the test rejects f with probability that is close to $3^{k+1}\delta$. This follows from the fact that in this case, the probability over the selection of $y_1,\cdots,y_{k+1},b$, that among the 3^{k+1} points $\sum_i c_i y_i + b$, the functions f and g differ in precisely one point, is close to $3^{k+1}\cdot\delta$. Observe that if they do, then the test rejects.

Lemma 3.5.20 *Suppose $0 \leq \eta_i \leq \frac{1}{2(2k+7)3^{k+1}}$. Let δ denote the relative distance between f and g, $\ell = 3^{k+1}$, and $Q \stackrel{def}{=} (\frac{1-\ell\delta}{1+\ell\delta})\cdot\ell\delta$. Then, when $y_1,\cdots,y_{k+1},b$ are chosen randomly, the probability that for exactly one point v among the ℓ points $\sum_i C_i y_i + b$, $f(v)\neq g(v)$ is at least Q.*

[21] Recall that we are using the convention $0^0 = 1$.

Observe that $\eta_i = \Omega(Q) = \Omega(3^{k+1}\delta)$.

Proof of Lemma 3.5.20: For each $C \in \mathbb{F}_3^{k+1}$, let X_C be the indicator random variable whose value is 1 if and only if $f(C \cdot Y + b) \neq g(C \cdot Y + b)$. Clearly, $\Pr[X_C = 1] = \delta$ for every C. It follows that the random variable $X = \sum_C X_C$ which counts the number of points v of the required form in which $f(v) \neq g(v)$ has expectation $\mathrm{E}[X] = 3^{k+1}\delta = \ell \cdot \delta$. It is not difficult to check that the random variables X_C are pairwise independent, since for any two distinct C_1 and C_2, the sums $\sum_{i=1}^{k+1} C_{1,i} + b$ and $\sum_{i=1}^{k+1} C_{2,i} + b$ attain each pair of distinct values in $\mathbb{F}_3^n$ with equal probability when the vectors are chosen randomly and independently. Since X_C's are pairwise independent, $\mathrm{Var}[X] = \sum_C \mathrm{Var}[X_C]$. Since X_C's are boolean random variables, we note

$$\mathrm{Var}[X_C] = \mathrm{E}[X_C^2] - (\mathrm{E}[X_C])^2 = \mathrm{E}[X_C] - (\mathrm{E}[X_C])^2 \leq \mathrm{E}[X_C].$$

Thus we obtain $\mathrm{Var}[X] \leq \mathrm{E}[X]$, so $\mathrm{E}[X^2] \leq \mathrm{E}[X]^2 + \mathrm{E}[X]$. Next we use the following inequality from [AKK+03] which holds for a random variable X taking nonnegative, integer values,

$$\Pr[X > 0] \geq \frac{(\mathrm{E}[X])^2}{\mathrm{E}[X^2]}.$$

In our case, this implies

$$\Pr[X > 0] \geq \frac{(\mathrm{E}[X])^2}{\mathrm{E}[X^2]} \geq \frac{(\mathrm{E}[X])^2}{\mathrm{E}[X] + (\mathrm{E}[X])^2} = \frac{\mathrm{E}[X]}{1 + \mathrm{E}[X]}.$$

Therefore,

$$\mathrm{E}[X] \;\geq\; \Pr[X = 1] + 2\Pr[X \geq 2] \;=\; \Pr[X = 1] + 2\left(\frac{\mathrm{E}[X]}{1 + \mathrm{E}[X]} - \Pr[X = 1]\right)$$

$$= \frac{2\mathrm{E}[X]}{1 + \mathrm{E}[X]} \;-\; \Pr[X = 1].$$

After simplification we obtain,

$$\Pr[X = 1] \geq \frac{1 - \mathrm{E}[X]}{1 + \mathrm{E}[X]} \cdot \mathrm{E}[X].$$

The proof is complete by recalling that $\mathrm{E}[X] = \ell \cdot \delta$. ■

Proof of Theorem 3.5.3: Clearly if f belongs to $\mathcal{P}_t$, then by Claim 3.3.1 the tester accepts f with probability 1.

Therefore let $\delta(f, \mathcal{P}_t) \geq \epsilon$. Let $d = \delta(f, g_r)$, where r is as in algorithm **Test-$\mathcal{P}_t$**. If $\eta < \frac{1}{2(2k+7)3^{k+1}}$ then by Lemma 3.5.15 $g_r \in \mathcal{P}_t$ and, by Lemma 3.5.20, $\eta_i = \Omega(3^{k+1} \cdot d) = \Omega(3^{k+1}\epsilon)$. Hence $\eta_i \geq min\left(\Omega(3^{k+1}\epsilon), \frac{1}{2(2k+7)3^{k+1}}\right)$. ■

Remark 3.5.21 *Theorem 3.5.1 follows from a similar argument.*

A basis of GRM consisting of minimum-weight codewords was considered in [DGM70, DK00]. We extend their result to obtain a different exact characterization for low-degree polynomials. Furthermore, it seems that their exact characterization can be turned into a robust characterization following analysis similar to our robust characterization, though we have not worked out the details. However, our basis is cleaner and yields a simpler analysis.

We point out that for degree smaller than the field size, the exact characterization obtained from [DGM70, DK00] coincides with [BLR93, RS96, FS95]. This provides an alternate proof to the exact characterization of [FS95] (for more details, see Remark 3.3.11 later and [FS95]).

Independently, Kaufman and Ron, generalizing a characterization result of [FS95], gave a tester for low degree polynomials over general finite fields (see [KR04]). They show that a given polynomial is of degree at most t *if and only if* the restriction of the polynomial to every affine subspace of suitable dimension is of degree at most t. Following this idea, their tester chooses a random affine subspace of a suitable dimension, computes the polynomial restricted to this subspace, and verifies that the coefficients of the higher degree terms are zero[22]. To obtain constant soundness, the test is repeated many times. An advantage of our approach is that in one round of the test (over the prime field) we test only one linear constraint, whereas their approach needs to test multiple linear constraints. We next explore local testability of Reed-Muller codes over general finite fields.

[22]Since the coefficients can be written as linear sums of the evaluations of the polynomial, this is equivalent to check several linear constraints

3.6 A Lower Bound and Improved Self-correction

3.6.1 A Lower Bound

The next theorem is a simple modification of a theorem in [AKK+03] and essentially implies that our result (over prime fields) is almost optimal. We restrict ourselves to prime fields, though the theorem and the proof can easily be extended over general finite fields.

Proposition 3.6.1 *Let $\mathcal{F}$ be any family of functions $f : \mathbb{F}_p^n \to \mathbb{F}_p$ that corresponds to a linear code $\mathcal{C}$. Let d denote the minimum distance of the code $\mathcal{C}$ and let $d^\perp$ denote the minimum distance of the dual code of $\mathcal{C}$.*
Every one-sided testing algorithm for the family $\mathcal{F}$ must perform $\Omega(d^\perp)$ queries, and if the distance parameter ϵ is at most d/p^{n+1}, then $\Omega(1/\epsilon)$ is also a lower bound for the necessary number of queries.

Lemma 3.3.2 and Proposition 3.6.1 gives us the following corollary.

Corollary 3.6.2 *Every one-sided tester for testing $\mathcal{P}_t$ with distance parameter ϵ must perform $\Omega(max(\frac{1}{\epsilon}, (1 + ((t+1) \ mod \ (p-1)))p^{\frac{t+1}{p-1}}))$ queries.*

For completeness, we include a brief proof of Theorem 3.6.1.

Proof of Theorem 3.6.1: The proof here is essentially from [AKNS99]. It is clear that $\Omega(d^\perp)$ queries are necessary (see [MS77], Chapter 5, Theorem 8). Next consider the case when $\epsilon < d/p^{n+1}$. In this case, the lower bound follows from an application of Yao's principle. We define two distributions, one of positive instances, and the other of negative instances. We then argue that in order to distinguish those distributions any algorithm must perform $\Omega(1/\epsilon)$ queries to achieve a success probability at least $2/3$. Let the positive distributions have all its mass at the zero vector $\vec{0} = \langle 0, \cdots, 0 \rangle$. For the negative distribution, we partition the set of all coordinates into $t = 1/\epsilon$ nearly equal parts $I_1, \cdots, I_t$ and give a weight $1/t$ to each of the characteristic vectors w_i of $I_i, i = 1, \cdots, t$. Notice $\vec{0} \in \mathcal{C}$ and that $\text{dist}(w_i, \mathcal{C}) = \epsilon$ due to the assumption on the minimum distance on $\mathcal{C}$. Finally, a random instances is generated by first choosing one of the distributions with probability $1/2$ and then generating a vector according to the chosen distribution. Assume that the hypothetical algorithm checks s bits. Since hypothetical algorithm is one-way, it accepts $\vec{0}$. Since it tests only s bits, it surely accepts at least $(t - s)$

negative instances. Therefore, it gives an incorrect answer with probability at least $(t-s)/2t < 1/3$. Therefore $s = \Omega(1/\epsilon)$. ■

3.6.2 Improved Self-correction

From Lemmas 3.5.4, 3.5.6 and 3.5.15 the following corollary is immediate:

Corollary 3.6.3 *Consider a function $f : \mathbb{F}_3^n \to \mathbb{F}_3$ that is ϵ-close to a degree-t polynomial $g : \mathbb{F}_3^n \to \mathbb{F}_3$, where $\epsilon < \frac{1}{2(2k+7)3^{k+1}}$. (Assume $k \geq 1$.) Then the function f can be self-corrected. That is, for any given $x \in \mathbb{F}_3^n$, it is possible to obtain the value $g(x)$ with probability at least $1 - 3^{k+1}\epsilon$ by querying f on 3^{k+1} points on $\mathbb{F}_3^n$.*

An analogous result may be obtained for the general case. We, however, improve the above corollary slightly. The above corrector does not allow any error in the 3^{k+1} points it queries. We obtain a stronger result by querying on a slightly larger flat H, but allowing some errors. Errors are handled by decoding the induced Generalized Reed-Muller code on H.

Proposition 3.6.4 *Consider a function $f : \mathbb{F}_p^n \to \mathbb{F}_p$ that is ϵ-close to a degree-t polynomial $g : \mathbb{F}_p^n \to \mathbb{F}_p$. Then the function f can be self-corrected. That is, assume $K > (k+1)$, then for any given $x \in \mathbb{F}_p^n$, the value of $g(x)$ can be obtained with probability at least $1 - \frac{\epsilon}{(1-\epsilon \cdot p^{k+1})^2} \cdot p^{-(K-2k-3)}$ with p^K queries to f.*

Proof: Our goal is to correct the $GRM_p(t,n)$ at the point x. Assume $t = (p-1)\cdot k + R$, where $0 \leq R \leq (p-2)$. Then the relative distance of the code δ is $(1 - R/p)p^{-k}$. Note that $2p^{-k-1} \leq \delta \leq p^{-k}$. Recall that the local testability test requires a $(k+1)$-flat, i.e., it tests $\sum_{c_1,\cdots,c_{k+1} \in \mathbb{F}_p} c_1^{p-2-R} f(y_0 + \sum_{i=1}^{k+1} c_i y_i) = 0$, where $y_i \in \mathbb{F}_p^n$.

We choose a slightly larger flat, i.e., a K-flat with $K > (k+1)$ to be chosen later. We consider the code restricted to this K-flat with point x being the origin. We query f on this K-flat. It is known that a majority logic decoding algorithm exists that can decode Generalized Reed-Muller code up to half the minimum distance for any choice of parameters (see [Sud01]). Thus if the number of error is small we can recover $g(x)$.

Let the relative distance of f from the code be ϵ and let S be the set of points where it disagrees with the closest codeword. Let the random K-flat be $H = \{x + \sum_{i=1}^{K} t_i u_i | t_i \in \mathbb{F}, u_i \in_R \mathbb{F}_p^n\}$.

Let the random variable $Y_{\langle t_1,\cdots,t_K\rangle}$ take the value 1 if $x+\sum_{i=1}^{K} u_i t_i \in S$ and 0 otherwise. Let $D = \mathbb{F}^K \setminus \{0\}$ and $U = \langle u_1, \cdots, u_K\rangle$. Define $Y = \sum_{\langle t_1,\cdots,t_K\rangle \in D} Y_{\langle t_1,\cdots,t_K\rangle}$ and $\ell = (p^K - 1)$. We would like to bound the probability

$$\Pr_U[|Y - \epsilon\ell| \geq (\delta/2 - \epsilon)\ell].$$

Since $\Pr_U[Y_{t_1,\cdots,t_K} = 1] = \epsilon$, by linearity we get $\mathrm{E}_U[Y] = \epsilon\ell$. Let $T = \langle t_1, \cdots, t_K\rangle$. Now

$$Var[Y] = \sum_{T\in\mathbb{F}^K-\{0\}} Var[Y_T] + \sum_{T\neq T'} Cov[Y_T, Y_{T'}] = \ell(\epsilon-\epsilon^2) + \sum_{T\neq\lambda T'} Cov[Y_T, Y_{T'}]$$

$$+ \sum_{T=\lambda T';1\neq\lambda\in\mathbb{F}^*} Cov[Y_T, Y_{T'}] \leq \ell(\epsilon-\epsilon^2) + \ell\cdot(p-2)(\epsilon-\epsilon^2) \;=\; \ell(\epsilon-\epsilon^2)(p-1).$$

The above follows from the fact that when $T \neq \lambda T'$ then the corresponding events Y_T and $Y_{T'}$ are independent and therefore $Cov[Y_T, Y_{T'}] = 0$. Also, when Y_T and $Y_{T'}$ are dependent then $Cov[Y_T, Y_{T'}] = \mathrm{E}_U[Y_T Y_{T'}] - \mathrm{E}_U[Y_T]\mathrm{E}_U[Y_{T'}] \leq \epsilon - \epsilon^2$. Therefore, by Chebyshev's inequality we have (assuming $\epsilon < p^{-(k+1)}$)

$$\Pr_U[|Y - \epsilon\ell| \geq (\delta/2 - \epsilon)\ell] \leq \frac{\ell\epsilon(1-\epsilon)(p-1)}{(\delta/2-\epsilon)^2\ell^2}$$

Now note $(\delta/2 - \epsilon) \geq (p^{-k-1} - \epsilon) = (1 - \epsilon\cdot p^{k+1})p^{-k-1}$. We thus have

$$\Pr_U[|Y - \epsilon\ell| \geq (\delta/2 - \epsilon)\ell] \leq \frac{\epsilon(1-\epsilon)(p-1)}{(1-\epsilon\cdot p^{k+1})^2 p^{-2k-2}\ell}$$

$$\leq \frac{\epsilon p}{(1-\epsilon\cdot p^{k+1})^2 p^{-2k-2}(\ell+1)} \;=\; \frac{\epsilon}{(1-\epsilon\cdot p^{k+1})^2}\cdot p^{-(K-2k-3)}.$$ ∎

3.7 Testing at a Large Distance

We prove a conditional inverse theorem for the fourth Gowers uniformity norm of boolean functions. That is, we show that if the fourth Gowers norm of a function is bounded away from zero, then the function has nontrivial correlation with a polynomial of degree at most three, on condition that a proposed bilinear testing can be extended to the low-end setting. In brief, we make the following assumption. If a function passes a certain bilinearity test restricted to some set A with constant

probability, then there exists a refinement A' of A (i.e., $A' \subseteq A$ and size of A' is at least a constant fraction of A), and a bilinear function such that the function agrees with the bilinear function everywhere on A'.

More generally, we show that if the $(d+1)^{th}$ Gowers uniformity norm of a boolean function is bounded away from zero, then the function has non-trivial correlation with a degree d polynomial, on condition to the availability of a certain low-end tester for multilinear functions.

3.7.1 Introduction

We have mentioned previously that Blum, Luby, and Rubinfeld designed the first algorithm to test whether a given function is linear [BLR93]. Given a truly linear function, it is easy to show that their algorithm always accepts. The non-trivial part is to show that whenever the algorithm accepts a function with high probability, then the function has large correlation with a linear function. Later Fourier theoretic analysis of their algorithm by Aumann et al. [AHRS99] produced a tighter and sharp result. In particular, they were able to correlate the soundness of the test to the largest Fourier coefficient of the function given.

As mentioned previously, Alon et al. [AKK$^+$03] have described an algorithm to test whether a given (multivariate) function over $\mathbb{F}_2$ has low degree (alse see [JPRZ04]). Their test can be interpreted in the framework of Gowers uniformity norm [Gow01]. In what follows, we restrict ourselves to $\mathbb{F}_2$ and write the additive group of $\mathbb{F}_2$ multiplicatively. In other words, we will identify $0 \mapsto 1$ and $1 \mapsto (-1)$. We begin with defining Gowers norm[23].

Definition 3.7.1 *For a function $f : \mathbb{F}_2^n \to \mathbb{R}$, the d^{th} order Gowers uniformity norm is defined to be the quantity*

$$\|f\|_{U^d} \stackrel{def}{=} \left(\mathbb{E}_{x,x_1,\cdots,x_d} \left[\prod_{J \subseteq \{1,\cdots,d\}} f\left(x + \sum_{i \in J} x_i \right) \right] \right)^{1/2^d},$$

where $x, y_1, \cdots, y_n$ are chosen uniformly and independently.

[23] For notational convenience, we restrict ourselves to the real domain.

Recall that the algorithm of Alon et al. [AKK$^+$03] estimates the expectation

$$\mathbb{E}_{x,x_1,\cdots,x_d}\left[\sum_{J\subseteq\{1,\cdots,d\}} f\left(x+\sum_{i\in J} x_i\right)\right]$$

Denote this expectation by p, that is with probability p over the choice of $x, y_1, \cdots, y_d$, function f fails the test and succeeds with probability $(1-p)$. Clearly then

$$\|f\|_{U^d}^{2^d} = (-1)\cdot p + 1\cdot(1-p) = 1-2p \implies p = \frac{1-\|f\|_{U^d}^{2^d}}{2}.$$

Rephrasing in these terms, the result of Alon et al. essentially says that if the d^{th} Gowers uniformity norm is close to 1, roughly at least $\left(1-O(d^{-1}2^{-d})\right)^{1/2^d}$, then the function is very close to a $(d-1)$ degree polynomial.

When $d=1$, i.e., for linear functions (or homomorphisms), Fourier analysis yields that if the corresponding test accepts some given function with probability $1/2+\epsilon$, then the function has non-trivial correlation with a linear function. Specifically it is easy to prove the following Proposition[24].

Proposition 3.7.2 *(Inverse theorem for $U^2(\cdot)$ norm) Let $f:\mathbb{F}_2^n\to\mathbb{R}$ be a bounded function. Then*

$$\sup|\hat{f}_\alpha| \;\le\; \|f\|_{U^2} \;\le\; \sup|\hat{f}_\alpha|^{1/2}.$$

Green and Tao [GT05] prove an inverse theorem for quadratic functions for vector spaces over prime fields with odd characteristics. Samorodintsky [Sam07] establishes an inverse theorem for the third Gowers uniformity norm for the boolean functions. We prove conditional inverse theorems for Gowers uniformity norm of order $O(1)$ in the boolean domain. In all these works, the key role is played by a quantitative version of the Balog-Szemeredi theorem proved in [Gow98, Gow01]. It also requires an analogue of Frienman's theorem over a finite field, a result due to Rusza.

In [Sam07], the inverse theorem is used to provide a tighter analysis for testing polynomials of degree at most two. In [ST06], Gowers uniformity norm is used to construct better PCPs (see [ALM$^+$98, AS98]). In [VW07] Gowers norm is used to give a new xor lemma.

In [Gow01], Gowers introduces the uniformity norm and provides a very weak

[24]The proposition holds even for an arbitrary abelian group.

converse that suffices to obtain an alternate proof of the Szemeridi's theorem. In there, it is (implicitly) conjectured that a function having a large $(d+1)^{th}$ Gowers uniformity norm should be correlated with a degree d polynomial.

Call a function $f : \mathbb{F}_2^n \times \mathbb{F}_2^n \to \mathbb{F}_2^n$ bilinear if it holds that $f(x,y) + f(x,z) = f(x, y+z)$ and $f(y,x)+f(z,x) = f(y+z,x)$ for all $x, y, z \in \mathbb{F}_2^n$. Suppose we are given a function $f : \mathbb{F}_2^n \times \mathbb{F}_2^n \to \mathbb{F}_2^n$. How can we test whether the function is a bilinear? A plausible approach would be to generalize the BLR test in the following way. Define $f_{,z}(u,v) \stackrel{\text{def}}{=} f(u, v+z) + f(u,v)$ and similarly, $f_{z,}(u,v) \stackrel{\text{def}}{=} f(u+z, v) + f(u,v)$. Then it is easy to see that a function is bilinear iff $f_{x,y}(u,v) = f(x,y)$. (Recall that in this notation, BLR can be stated as testing $f_x(y) = f(x)$.) A slight generalization, as observed by Gowers [Gow01], would be to test whether

$$f(w,h) = f(x,y) + f(x, y+h) + f(x+w, z) + f(x+w, z+h)$$

holds for all $x, y, z, w, h \in \mathbb{F}_2^n$. Following an analysis similar to [BOCLR04] the test can be made robust. That is, if a function is accepted with probability higher than $23/25$, then the function can be shown to be very close to a bilinear function.

In [Sam07] it is shown that a function passing the BLR test with probability $\Omega(1)$ has non-trivial correlation with a linear function. (In fact, his result is even more general and works even over any Abelian p-group.) The central idea in his work follows from a theorem due to Gowers [Gow01]. It roughly says (qualitatively) the following: Let A be an arbitrary subset of an abelian group such that $A+A$ has $\Omega(|A|^2)$ many collisions, that is, distinct solution to the linear equation $x+y = z+w$. Then there exists a subset $A' \subset A$ of size $\Omega(|A|)$ which is approximately closed additively, meaning $|A' + A'| = O(|A'|)$. If a function f passes the BLR tests with non-negligible probability, then it can be shown that there exists a large set A, such that $(A, f(A))$ produces many collisions. Following Gowers, this then implies a large set $(A', f(A'))$ which is approximately-closed additively. Finally, a generalization of Friemans's theorem due to Rusza allows him to roughly correlate the function with an affine function on a constant fraction of the point on A'.

Here we establish a connection between testing multilinearity and testing degree d polynomials. We show that the approach in [Sam07] can be generalized provided a multilinearity test can be extended to the lower-end.

3.7.2 A Characterization of Low-degree Polynomials over Prime Fields

A natural question that arises is why one would expect Gowers norm to play any role in determining whether a polynomial is of low degree. To see this, observe that the Gowers norm (raised to appropriate the power) of order d can be viewed as a d^{th} order "discrete derivative" of the function. We outline the following (perhaps folklore) proposition.

Proposition 3.7.3 *Let $\mathbb{F} = \mathbb{F}_p$ for some prime p. Let $f \in \mathbb{F}[x_1, \cdots, x_n]/(x_1^p - x_1, \cdots, x_n^p - x_n)$ (that is each individual degree is at most $p-1$). Then degree$(f) \leq d$* iff *for all $x, y_1, ..., y_k \in \mathbb{F}^n, k = d+1$,*

$$\sum_{S \subseteq [k]} (-1)^{|S|} f\left(x + \sum_{j \in S} y_j\right) = 0.$$

This is indeed a characterization and follows from the following simple lemma (and induction).

Lemma 3.7.4 *Let $f \in \mathbb{F}[x_1, \cdots, x_n]/(x_1^p - x_1, \cdots, x_n^p - x_n)$ of degree exactly d. Then there is a choice of $y \in \mathbb{F}^n$ such that $f(x+y) - f(x)$ is exactly a degree $d-1$ polynomial in xs, and for all choices of y, the degree of $f(x+y) - f(x)$ is at most $d-1$.*

Proof: Denote $f_y(x) \stackrel{\text{def}}{=} f(x+y) - f(x)$. It is clear that the degree of $f_y(x)$ can be at most $d-1$ for all choices of y. Therefore, it suffices to prove the existence of an y that makes the degree of $f_y(x)$ exactly $d-1$. Let

$$f(x) = \sum_{I = \langle I_1, \cdots, I_n \rangle} c_I \prod_j x_j^{I_j}.$$

Then,

$$f_y(x) = \sum_{I = \langle I_1, \cdots, I_n \rangle} c_I \left(\prod_j (x_j + y_j)^{I_j} - \prod_j x_j^{I_j} \right).$$

Treat $f_y(x)$ as a formal polynomial. Since $\mathbb{F}$ is a prime field, observe that no binomial coefficients vanishes. Collect all the terms linear in y which is

$$\sum_j y_j \underbrace{\left(\sum_I c_I I_j x_j^{I_j - 1} \prod_{k \neq j} x_k^{I_k} \right)}_{X_j}.$$

Observe that for each y_j, X_j is a non-zero polynomial if and only if f depends on x_j. In particular, there exists at least one j, say j^*, for which X_{j^*} is a non-zero polynomial of degree exactly $(d-1)$. Choosing y with $y_{j^*} = 1$, and for all $j \neq j^*, y_j = 0$ completes the proof. ■

3.7.3 A Conditional Inverse Theorem for the Gowers Norm of Order Four

Unless otherwise stated, we work over[25] $\mathbb{F}_2$. We use notation introduced in Chapter 2 (cf. Section 2.1.3). Matrices are very useful to define linear functions. Here, we deal with bilinear (and multilinear) functions. To handle these functions, we use tensors throughout.

We mention a simple fact, observed in [Sam07].

Fact 3.7.5 *For a boolean function f, it holds $f_x * f_x(s) = f_s * f_s(x)$.*

Proof:

$$f_x * f_x(s) = \mathbb{E}_y f_x(y) f_x(y+s) = \mathbb{E}_y f(x+y) f(y) f(x+y+s) f(y+s)$$

$$= \mathbb{E}_y f_s(x+y) f_s(y) = f_s * f_s(x).$$

■

Theorem 3.7.6 *Let $f : \{0,1\}^n \to \{-1,1\}$ be a function such that $\|f\|_{U_4} \geq \epsilon$. Assuming Conjecture 3.7.11 holds, there exists a cubic polynomial g such that the distance between f and g is at most $\frac{1}{2} - \epsilon'$, where $\epsilon' = \text{poly}\left(\epsilon, \xi(\epsilon^{O(1)})\right)$.*

[25] We will mostly use the additive abelian group. For notational convenience, we may sometime write this group multiplicatively.

In what follows, whenever we use ϵ (or ϵ' etc.), we mean $\epsilon = \Omega(1)$.

To prove Theorem 3.7.6, we closely follow the approach developed by Samorodnitsky [Sam07]. As in there, we break the proof into two parts. First we show (the easier part) that if a certain expectation involving the second order derivative of f is large, then f is actually close to a cubic function. Later, we establish (the difficult part) that if the Gowers fourth norm is indeed large, then that certain expectation is large.

Let us begin with a cubic function f. Then observe that f can be written so that

$$Łf(x) = \sum_i x_i \langle x, \mathsf{M}_i x \rangle \stackrel{\text{def}}{=} \langle x, x \cdot \mathsf{M} x \rangle,$$

where $\mathsf{M} = \{\mathsf{M}_i\}_{i \in [n]}$, for some binary matrices M_i. Now consider the first derivative of the f

$$Łf_y(x) = \langle y, x \cdot \mathsf{M} x \rangle + \langle x, y \cdot \mathsf{M} x \rangle + \langle x, x \cdot \mathsf{M} y \rangle + \langle y, y \cdot \mathsf{M} x \rangle + \langle y, x \cdot \mathsf{M} y \rangle + \langle x, y \cdot \mathsf{M} y \rangle + a_y,$$

where a_y is a constant (depending upon M and y). Define $\mathsf{N}_{jki} \stackrel{\text{def}}{=} \mathsf{M}_{ijk}$. With this definition

$$\langle y, x \cdot \mathsf{M} x \rangle = \sum_{ijk} x_i y_j \mathsf{M}_{ijk} x_k = \sum_{ijk} y_j x_k \mathsf{N}_{jki} x_j = \langle x, y \cdot \mathsf{N} x \rangle$$

Similarly defining $\mathsf{L}_{kij} \stackrel{\text{def}}{=} \mathsf{M}_{ijk}$, we get $\langle x, x \cdot \mathsf{M} y \rangle = \langle x, y \cdot \mathsf{L} x \rangle$. Thus setting $\mathsf{P} \stackrel{\text{def}}{=} \mathsf{M} + \mathsf{N} + \mathsf{L}$, we get

$$Łf_y(x) = \langle x, y \cdot \mathsf{P} x \rangle + \langle y, y \cdot \mathsf{P} x \rangle + a_y = \sum_i y_i \left[\langle x, \mathsf{P}_i x \rangle + \langle y, \mathsf{P}_i x \rangle \right] + a_y$$

Note that

$$\mathsf{P}_{ijk} = \mathsf{M}_{ijk} + \mathsf{N}_{ijk} + \mathsf{L}_{ijk} = \mathsf{M}_{ijk} + \mathsf{M}_{kij} + \mathsf{M}_{jki} = \mathsf{P}_{jki} = \mathsf{P}_{kij}.$$

Similarly $\mathsf{P}_{ikj} = \mathsf{P}_{jik} = \mathsf{P}_{kji}$. Define $\Gamma_i = \mathsf{P}_i + \mathsf{P}_i^t$. Observe that Γ is invariant under the action of group Sym_3, where the group acts by permuting the indices of Γ. Further observe that $\Gamma_{ijj} = 0$.

We now consider the second derivative of f. Clearly

$$Łf_{yu}(x) = \sum_i y_i \langle x, \Gamma_i u \rangle + a_{yu} = \langle x, y \cdot \Gamma u \rangle + a_{yu},$$

where $a_{yu} = \langle u + y, y \cdot \mathsf{P} u \rangle$.

Ignoring the phase for the moment (or equivalently changing the signs if necessary), it is clear that if f is a degree at most three polynomial then $\hat{f}^2(y \cdot \Gamma u) = 1$ for some Γ with nice properties.

This gives the motivation for the first part. We now show that a somewhat converse to the above actually holds.

Lemma 3.7.7 *Let Γ be a tensor of rank three and that it is invariant under the action of Sym_3 group. Further assume $\Gamma_{ijj} = 0$ for all $i, j \in [n]$. If*

$$\mathbb{E}_{uy} \hat{f}^2_{uy}(y \cdot \Gamma u) \geq \epsilon_1,$$

Then there exists a cubic polynomial h such that

$$\|f - h\| \leq \frac{1}{2} - \epsilon'.$$

Proof: First from Γ, recover M, i.e., for each $i, j, k \in [n]$ (such that $k \neq i \neq j \neq k$), set

$$\mathsf{M}_{ijk} + \mathsf{M}_{ikj} + \mathsf{M}_{jik} + \mathsf{M}_{jki} + \mathsf{M}_{kij} + \mathsf{M}_{kji} = \Gamma_{ijk},$$

and set $\mathsf{M}_{ijj} = \mathsf{M}_{jij} = \mathsf{M}_{jji} = 0$ (for all i, j). Then define $g(x) = (-1)^{\langle x, x \cdot \mathsf{M} x \rangle}$. Then clearly $g_{yu}(x) = (-1)^{\langle x, y \cdot \Gamma u \rangle}$, up to a phase. Therefore,

$$\mathbb{E}_{uy} [\, \mathbb{E}_z \, f_{yu}(z) g_{yu}(z) \,]^2 = \mathbb{E}_{uy} \hat{f}^2_{uy}(y \cdot \Gamma u) \geq \epsilon_1$$

However, Lemma A.2.9 shows that

$$\mathbb{E}_{uy} [\, \mathbb{E}_z \, f_{yu}(z) g_{yu}(z) \,]^2 = \sum_\alpha \widehat{fg}^4_\alpha \leq \max_\alpha \widehat{fg}^2_\alpha. \tag{3.37}$$

Let β be a vector such that $|\widehat{fg}_\beta| \geq \sqrt{\epsilon_1}$. This then implies that there is a choice of

$c \in \{0,1\}$ such that for a cubic polynomial $h(x) \stackrel{\text{def}}{=} (-1)^{\langle x, x \cdot \mathsf{M} x\rangle + \langle x, \beta\rangle + c}$ it holds that

$$\|f - h\| \leq \frac{1}{2} - \epsilon'.$$

■

Therefor to prove Theorem 3.7.6, it suffice to show that $\|f\|_{U^4} \geq \epsilon \implies \exists \Gamma$ with nice properties such that $\mathbb{E}_{yu} \hat{f}^2_{uy}(y \cdot \Gamma u) \geq \epsilon_1$.

Lemma 3.7.8 *It holds that*

$$\|f\|^{16}_{U^4} = \mathbb{E}_{uy} \sum_{\alpha} \hat{f}^4_{uy}(\alpha)$$

Proof:

$$\|f\|^{16}_{U^4} = \mathbb{E}_{xyzuw} f(x) \cdots f(x+y+z+w+u) = \mathbb{E}_u \mathbb{E}_{xyzw} f_u(x) \cdots f_u(x+y+z+w)$$

$$= \mathbb{E}_u \|f_u\|^8_{U^3} = \mathbb{E}_{uy} \sum_{\alpha} \hat{f}^4_{uy}(\alpha),$$

where we used Lemma A.2.1. ■

Claim 3.7.9 *If $\|f\|_{U^4} \geq \epsilon$, then $\mathbb{E}_{uy} \sum_\alpha \hat{f}^6_{uy}(\alpha) \geq \epsilon^{32}$.*

Proof: Follows from Holder's inequality, i.e., using $(\sum_a a^4) \leq (\sum_a a^2)^{1/2} (\sum_a a^6)^{1/2}$. ■

Define a product distribution on (symmetric) functions $\phi : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ by defining $\Pr[\phi(y,u) = \alpha] = \hat{f}^2_{yu}(\alpha)$. (Note that $\phi(u,y) = \phi(y,u)$ holds for all (u,y).) Further the choices for distinct pairs of u and y are independent. Let $\rho = \Omega(1)$ be chosen later.

By a vertical parallelogram P of height h and width w, we mean $\{(x,y), (x, y+h), (x+w, z), (x+w, z+h)\}$. We also extend our function ϕ to be defined on a parallelogram P :

$$\phi(P) \stackrel{\text{def}}{=} \phi(x,y) + \phi(x, y+h) + \phi(x+w, z) + \phi(x+w, z+h).$$

For a vertical parallelogram P, we define its width $w(P)$ and height by $h(P)$. We use P^{xyz}_{hw} to denote parallelogram $\{(x,y), (x, y+h), (x+w, z), (x+w, z+h)\}$ of width w and height h.

Then define the random variable L on this probability space as follows:

$$L(\phi) \stackrel{\text{def}}{=} \Pr_{h,w,x,y,z}\big[\,\phi(w,h) = \phi(x,y) + \phi(x,y+h) + \phi(x+w,z) + \phi(x+w,z+h);$$

$$\hat{f}^2_{wh}(\phi(w,h)) \geq \rho; \cdots ; \hat{f}^2_{(x+w)(z+h)}(\phi(x+w,z+h)) \geq \rho\Big]\,.$$

Lemma 3.7.10

$$\mathbb{E}_\phi L(\phi) \geq \epsilon_2$$

Proof:

$$\mathbb{E}_\phi(L(\phi)) = \mathbb{E}_{hwxyz}\Pr_\phi\left[\phi(w,h) = \phi(x,y) + \phi(x,y+h) + \phi(x+w,z) + \phi(x+w,z+h);\right.$$

$$\left.\hat{f}^2_{wh}(\phi(h,w)) \geq \rho; \cdots ; \hat{f}^2_{(x+w)(z+h)}(\phi(x+w,z+h)) \geq \rho\right]$$

$$= \mathbb{E}_{hwxyz} \sum_{\substack{\hat{f}^2_{wh}(\alpha+\beta+\gamma+\lambda)\geq\rho;\cdots;\\ \hat{f}^2_{(x+w)(z+h)}(\gamma)\geq\rho,\alpha,\beta,\gamma,\lambda}} \hat{f}^2_{hw}(\alpha+\beta+\gamma+\lambda)\hat{f}^2_{xy}(\alpha)\hat{f}^2_{x(y+h)}(\beta)\hat{f}^2_{(x+w)z}(\gamma)\hat{f}^2_{(x+w)(z+h)}(\lambda)$$

$$\geq \mathbb{E}_{hwxyz} \sum_{\alpha\beta\gamma\lambda} \hat{f}^2_{hw}(\alpha+\beta+\gamma+\lambda)\hat{f}^2_{xy}(\alpha)\hat{f}^2_{x(y+h)}(\beta)\hat{f}^2_{(x+w)z}(\gamma)\hat{f}^2_{(x+w)(z+h)}(\lambda) - 5\rho \tag{3.38}$$

$$= \mathbb{E}_{whxyz} \sum_{\alpha\beta\gamma\lambda} \mathbb{E}_{u_1,\cdots,u_5,s_1\cdots,s_5}\Big(\hat{f}_{xy}(u_1)\hat{f}_{xy}(u_1+s_1)\chi_\alpha(s_1)\hat{f}_{x(y+h)}(u_2)\hat{f}_{x(y+h)}(u_2+s_2)\chi_\beta(s_2)$$

$$\hat{f}_{(x+w)z}(u_3)\hat{f}_{(x+w)z}(u_3+s_3)\chi_\gamma(s_3)\hat{f}_{(x+w)(z+h)}(u_4)\hat{f}_{(x+w)(z+h)}(u_4+s_4)\chi_\lambda(s_4)$$

$$\hat{f}_{hw}(u_5)\hat{f}_{hw}(u_5+s_5)\chi_{\alpha+\beta+\gamma+\lambda}(s_5)\Big) - 5\rho$$

$$= \mathbb{E}_{whxyzs} f_{xy} * f_{xy}(s) f_{x(y+h)} * f_{x(y+h)}(s) f_{(x+w)z} * f_{(x+w)z}(s) f_{(x+w)(z+h)} * f_{(x+w)(z+h)}(s) f_{wh} * f_{wh}(s) - 5\rho$$

$$= \mathbb{E}_{whxyzs} f_{xs} * f_{xs}(y) f_{xs} * f_{xs}(y+h) f_{(x+w)s} * f_{(x+w)s}(z) f_{(x+w)s} * f_{(x+w)s}(z+h) f_{ws} * f_{ws}(h) - 5\rho$$

$$= \mathbb{E}_{whxyzs} \sum_{\alpha\beta\gamma\lambda\mu} \hat{f}^2_{xs}(\alpha)\chi_\alpha(y)\hat{f}^2_{xs}(\beta)\chi_\beta(y+h)\hat{f}^2_{(x+w)s}(\gamma)\chi_\gamma(z)\hat{f}^2_{(x+w)s}(\lambda)\chi_\lambda(z+h)\hat{f}^2_{ws}(\mu)\chi_\mu(h) - 5\rho$$

$$= \sum_{\alpha\beta\gamma\lambda\mu} \mathbb{E}_{wxs}\hat{f}^2_{sx}(\alpha)\hat{f}^2_{sx}(\beta)\hat{f}^2_{s(x+w)}(\gamma)\hat{f}^2_{s(x+w)}(\lambda)\hat{f}^2_{sw}(\mu)\mathbb{E}_y\chi_y(\alpha+\beta)\mathbb{E}_z\chi_z(\gamma+\lambda)\mathbb{E}_h\chi_h(\beta+\lambda+\mu) - 5\rho$$

$$= \sum_{\alpha\beta\gamma\lambda\mu} \mathbb{E}_{wxs}\hat{f}^2_{sx}(\alpha)\hat{f}^2_{sx}(\beta)\hat{f}^2_{s(x+w)}(\gamma)\hat{f}^2_{s(x+w)}(\lambda)\hat{f}^2_{sw}(\mu)\delta^\alpha_\beta\delta^\gamma_\lambda\delta^\beta_{\lambda+\mu} - 5\rho$$

$$= \mathbb{E}_{sxy} \sum_{\alpha\beta} \hat{f}^4_{sx}(\alpha)\hat{f}^4_{sy}(\beta)\hat{f}^2_{s(x+y)}(\alpha+\beta) - 5\rho \tag{3.39}$$

Denote $\epsilon_3 = \epsilon^{32}$. However, we are given

$$\mathbb{E}_{sxy} \sum_{\alpha\beta} \hat{f}^2_{sx}(\alpha)\hat{f}^2_{sy}(\beta)\hat{f}^2_{s(x+y)}(\alpha+\beta) = \mathbb{E}_{uy} \sum_{\alpha} \hat{f}^6_{uy}(\alpha) \geq \epsilon_3$$

where we use Corollary A.2.3 (Lemma A.2.2). By averaging argument

$$\Pr_{sxy}\left[\sum_{\alpha\beta} \hat{f}^2_{sx}(\alpha)\hat{f}^2_{sy}(\beta)\hat{f}^2_{s(x+y)}(\alpha+\beta) \geq \epsilon_3/2 \right] \geq \frac{\epsilon_3}{2}.$$

Applying Lemma A.2.6 and setting $\rho \stackrel{\text{def}}{=} \frac{\epsilon_3^6}{20\times 6^5}$, we get

$$\mathbb{E}_{\phi}(L(\phi)) \geq \mathbb{E}_{sxy} \sum_{\alpha\beta} \hat{f}^4_{sx}(\alpha)\hat{f}^4_{sy}(\beta)\hat{f}^2_{s(x+y)}(\alpha+\beta) - 5\rho \geq \frac{\epsilon_3^6}{4\cdot 6^5}. \tag{3.40}$$

Set $\epsilon_2 = \frac{\epsilon_3^6}{4\cdot 6^5} = \Omega(\epsilon^{192})$. ■

Let fix a ϕ for which $L(\phi) \geq \epsilon_2$. For this function we now define

$$A \stackrel{\text{def}}{=} \{(y,u) \mid \hat{f}^2_{uy}(\phi(u,y)) \geq \rho\}.$$

Clearly $|A| \geq \sqrt{\frac{\epsilon_2}{2}} \cdot 2^{2n} = \Omega\left(2^{2n}\right)$ holds for our choice of ϕ, since otherwise

$$L(\phi) \leq \frac{\sum_{\{(w,h)\}\cup P^{xyz}_{wh} \in A} 1}{\sum_{\{w,h\}\cup P^{xyz}_{wh} \in (\mathbb{F}_2^n)^2} 1} \leq \Pr_{whxyz}[\,(w,h),(x,y) \in A\,] < \frac{\epsilon_2}{2}.$$

Conjecture 3.7.11 *Let $\psi : \mathbb{F}_2^n \times \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a function and, $A \subseteq \mathbb{F}_2^n \times \mathbb{F}_2^n$ be a set such that $|A| = \Omega(2^{2n})$. Let P^{xyz}_{hw} denote parallelogram $\{(x,y),(x,y+h),(x+w,z),(x+w,z+h)\}$. It is given that*

$$\Pr_{\substack{wh\\xyz}}\left[\psi(w,h) = \psi(P^{xyz}_{hw}) \mid \{(w,h)\} \cup P^{xyz}_{wh} \subseteq A\right] \geq \epsilon.$$

Then[26] *there exists a refinement A' of A, i.e., $A' \subseteq A$ and $|A'| = \Omega(|A|)$, and a*

[26] If the error is smaller than $2/25$, then such a result can be shown to exist. See Appendix A.2.1.

bilinear function $\theta : \mathbb{F}_2^n \times \mathbb{F}_2^n \to \mathbb{F}_2^n$ *such that* $\psi|_{A'} = \theta|_{A'}$*, i.e., for all* $(x, y) \in A'$*,* $\psi(x, y) = \theta(x, y)$*. We set* $|A'| = \xi(\epsilon)|A|$*.*

We apply the above conjecture to our function ϕ and set A. Let θ be the ensured bilinear function and A' be its ensured refinement of size $\xi(\epsilon_2)|A|$. Moreover, it is easy to see that θ is symmetric since ϕ (and hence, A) is symmetric. We write $\theta(y, u) = y \cdot \mathsf{D}u$, where $\theta(e_i, e_j) = \mathsf{D}_{ij}$, i.e., j^{th} column of the i^{th} matrix. D is a tensor of rank three over $\mathbb{F}_2$. Combining all these observation, we get

$$\mathbb{E}_{yu}\hat{f}^2_{yu}(y \cdot \mathsf{D}u) \geq \sqrt{\frac{\epsilon_2}{2}} \cdot \xi(\epsilon_2) \cdot \rho \stackrel{\text{def}}{=} \epsilon_4,$$

and that $y \cdot \mathsf{D}u = u \cdot \mathsf{D}y$.

We first establish the following useful lemma.

Lemma 3.7.12 $\hat{f}_{uy}(z) = 0$ *for any* u, y, z *with* $\langle u + y, z\rangle = 1$*.*

Proof:

$$\hat{f}_{uy}(z) = \mathbb{E}_x f_{uy}(x)\chi_z(x) = \mathbb{E}_x f(x)f(x+u)f(x+y)f(x+u+y)\chi_z(x)$$

$$= \mathbb{E}_v f(v)f(v+y)f(u+v)f(u+v+y)\chi_z(u+v+y) =$$

$$\chi_z(u+y)\mathbb{E}_v f(v)f(v+y)f(u+v)f(u+v+y)\chi_z(v) = -\hat{f}_{uy}(z) \qquad \blacksquare$$

We now define the following set of functions, one for each fixed y: $g^y(u) \stackrel{\text{def}}{=} (-1)^{\langle y+u, y\cdot\mathsf{D}u\rangle}$. Note that

$$g^y * g^y(u) = \mathbb{E}_s g^y(s)g^y(s+u) = g^y(u)\mathbb{E}_s\chi_s(y \cdot \mathsf{D}u)\chi_s(y \cdot \mathsf{D}^t u) = g^y(u)\delta^{y\cdot\mathsf{D}u}_{y\cdot\mathsf{D}^t u}.$$

Now Lemma 3.7.12 yields the following equality.

$$\forall y \;\; \mathbb{E}_u g^y(u)\hat{f}^2_{uy}(y \cdot \mathsf{D}u) = \mathbb{E}_u \hat{f}^2_{uy}(y \cdot \mathsf{D}u).$$

On the other hand, letting $F^y(x) \stackrel{\text{def}}{=} \hat{f}^2_{yx}(y \cdot \mathsf{D}x)$ and using Claim A.2.4 we get

$$\forall y \;\; \mathbb{E}_u g^y(u)\hat{f}^2_{yu}(y\cdot\mathsf{D}u) = \mathbb{E}_u g^y(u)F^y(u) = \sum_z \widehat{g^y}(z)\widehat{F^y}(z) = \sum_z \widehat{g^y}(z)\mathbb{E}_u\hat{f}^2_{yu}(y\cdot\mathsf{D}^t u+z).$$

Since $\lambda_z = \mathbb{E}_u\hat{f}^2_{yu}(y \cdot \mathsf{D}^t u + z)$ are nonnegative and adds up to one, applying

Jensen's inequality (i.e., $\mathbb{E}[X^2] \geq (\mathbb{E}[X])^2$) we get

$$\forall y \quad \sum_z \widehat{g^y}^2(z)\mathbb{E}_u \hat{f}^2_{yu}(y \cdot \mathsf{D}^t u + z) \geq \left(\mathbb{E}_u g^y(u)\hat{f}^2_{yu}(y \cdot \mathsf{D}u)\right)^2$$

$$\implies \mathbb{E}_y \sum_z \widehat{g^y}^2(z)\mathbb{E}_u \hat{f}^2_{yu}(y \cdot \mathsf{D}^t u + z) \geq \mathbb{E}_y \left[\mathbb{E}_u g^y(u)\hat{f}^2_{yu}(y \cdot \mathsf{D}u)\right]^2$$

$$\geq \left(\mathbb{E}_{yu} g^y(u)\hat{f}^2_{yu}(y \cdot \mathsf{D}u)\right)^2 \geq \epsilon_4^2$$

However, it holds that

$$\sum_z \widehat{g^y}^2(z) \left(\mathbb{E}_u \hat{f}^2_{yu}(y \cdot \mathsf{D}^t u + z)\right) = \mathbb{E}_u (g^y * g^y)(u)\hat{f}^2_{yu}(y \cdot \mathsf{D}u) = \mathbb{E}_u \delta^{y \cdot Du}_{y \cdot D^t u} g_y(u)\hat{f}^2_{yu}(y \cdot \mathsf{D}u).$$

Thus taking expectation over y and simplifying

$$\mathbb{E}_{yu} \delta^{y \cdot Du}_{y \cdot D^t u} g_y(u)\hat{f}^2_{yu}(y \cdot \mathsf{D}u) = \mathbb{E}_{yu}\ \delta^{y \cdot Du}_{y \cdot D^t u}\ \hat{f}^2_{yu}(y \cdot \mathsf{D}u)\ \geq\ \epsilon_4^2.$$

Now for $y \in \{0,1\}^n$ define $S_y = \{u \mid y \cdot \mathsf{D}u = y \cdot \mathsf{D}^t u\}$. Set $S \stackrel{\text{def}}{=} \cup_y \{y\} \times S_y$. Note that $(y,u) \in S \implies (u,y) \in S$. Indeed observe that S is actually a symmetric subset of $\mathbb{F}_2^n \times \mathbb{F}_2^n$, and each S_y is a subspace of $\mathbb{F}_2^n$. Now define Λ following way : First define Λ on S by setting $y \cdot \Lambda u \stackrel{\text{def}}{=} y \cdot \mathsf{D}u$ on S. Next extend Λ bilinearly to the whole space.

Now observe that $\forall x, y, u$ and for any permutation $\sigma : \{x, y, u\} \to \{x, y, u\}$ it holds that $\langle x, y \cdot \Lambda u\rangle = \langle \sigma(x), \sigma(y) \cdot \Lambda\sigma(u)\rangle$. This ensures that we have

$$\mathbb{E}_{yu} \hat{f}^2_{yu}(y \cdot \Lambda u) \geq \epsilon_5 \stackrel{\text{def}}{=} \epsilon_4^2,$$

where Λ is invariant under the action of Sym_3 (see Claim A.2.7).

Therefore all we need to show is that $\Lambda_{ijj} = 0$. This requires some work. First define a set of functions, one for each y, $h^y(x) \stackrel{\text{def}}{=} (-1)^{\langle x, x \cdot \Lambda y\rangle}$. Let T be the matrix $T_{ij} \stackrel{\text{def}}{=} \Lambda_{ijj}$. Then we observe that

$$Łh^y(x) = \langle x, x \cdot \Lambda y\rangle = \langle x, Ty\rangle.$$

We need the corollary of the following lemma from [Sam07].

Lemma 3.7.13 *([Sam07]) Let f be a boolean function. If $\langle x, y\rangle = 1$, then $\hat{f}_y(x) = 0$.*

Corollary 3.7.14 *Let f be a boolean function. For all u, $\langle x, z\rangle = 1 \implies \hat{f}_{ux}(z) = 0$.*

By the above corollary, we have

$$\mathbb{E}_{xy} h^y(x) \hat{f}^2_{xy}(x \cdot \Lambda y) = \mathbb{E}_{xy} \hat{f}^2_{xy}(x \cdot \Lambda y).$$

Therefore, we have

$$\epsilon_5 \leq \mathbb{E}_{xy} h^y(x) \hat{f}^2_{yx}(x \cdot \Lambda y) = \mathbb{E}_{xy} (-1)^{\langle x, Ty\rangle} \hat{f}^2_{yx}(x \cdot \Lambda y) \leq \mathbb{E}_y 2^{-n} \sum_{x \perp Ty} \hat{f}^2_{yx}(x \cdot \Lambda y)$$

Note that $\langle x + y, x \cdot \Lambda y\rangle = \langle x, Ty\rangle + \langle y, Tx\rangle$. With this observation, an application of Lemma 3.7.12 yields

$$2^{-2n} \sum_{\substack{x,y, x \perp Ty \\ \langle x+y, x \cdot \Lambda y\rangle}} \hat{f}^2_{yx}(x \cdot \Lambda y) \geq \epsilon_5$$

Let $S \stackrel{\text{def}}{=} \{(x, y) \mid \langle x, Ty\rangle = \langle y, Tx\rangle = 0\}$. Further denote $Y_x = \{y | (x, y) \in S\}$. Clearly Y_x is a linear space. Let $\{(e_i, e_j) \mid i, j \in [n]\} \subseteq \mathbb{F}_2^n \times \mathbb{F}_2^n$ be a basis. Define Γ on the basis point as follows. For any (e_i, e_j) set $e_i \cdot \Gamma e_j \stackrel{\text{def}}{=} \Gamma(e_i, e_j)$ where

$$\Gamma(e_i, e_j) \stackrel{\text{def}}{=} \begin{cases} 0 & \text{if } e_i = e_j. \\ e_i \cdot \Lambda e_j & \text{if } (e_i, e_j) \in S \\ z \text{ s. t. } \langle z, e_i\rangle = \langle z, e_j\rangle = 0 \text{ and } \langle z, e_k\rangle = \langle e_k, e_i \cdot \Lambda e_j\rangle \; \forall k \in [n] \setminus \{i, j\}, & \text{ow} \end{cases}$$

Now extend Γ bilinearly everywhere in $\mathbb{F}_2^n \times \mathbb{F}_2^n$. Clearly Γ is symmetric. Moreover, for all x, y, it holds that $\langle x, x \cdot \Gamma y\rangle = 0 = \langle y, x \cdot \Gamma y\rangle$. To see this observe

$$\langle x, x \cdot \Gamma y\rangle = \sum_{i,j,k} x_i x_j x_k \langle e_i, e_j \cdot \Gamma e_k\rangle = \sum_k \sum_{i \neq j} x_i x_j x_k \left(\langle e_i, e_j \cdot \Gamma e_k\rangle + \langle e_j, e_i \cdot \Gamma e_k\rangle\right) = 0$$

Furthermore observe that $S = \cup_x \{x\} \times Y_x = \cup_x Y_x \times \{x\}$. Therefore, for all

$(x, y) \in S$ it holds that $x \cdot \Gamma y = x \cdot \Lambda y$ and hence,

$$\mathbb{E}_{yu} \hat{f}^2_{yu}(y \cdot \Gamma u) \geq \epsilon_5.$$

Proposition 3.7.15 *Γ is invariant under the action of Sym_3 and also, $\Gamma ijj = 0$. Further it holds that*

$$\mathbb{E}_{xy} \hat{f}^2_{xy}(x \cdot \Gamma y) \geq \epsilon_5.$$

Proof: First note that if for all x, y $\langle x, x \cdot \Gamma y \rangle = 0$, then $\Gamma_{ijj} = 0$. We already know $x \cdot \Gamma y = y \cdot \Gamma x$. Moreover, we know Γ is bilinear. It follows then that $\langle x, y \cdot \Gamma z \rangle = \langle y, x \cdot \Gamma z \rangle$ (i.e., Γ is invariant under the action of Sym_3). This is because

$$0 = \langle x+y, (x+y) \cdot \Gamma z \rangle = \langle x, y \cdot \Gamma z \rangle + \langle y, x \cdot \Gamma z \rangle \implies \langle x, y \cdot \Gamma z \rangle = \langle y, x \cdot \Gamma z \rangle \quad \blacksquare$$

This completes the proof of the theorem with the setting of $\epsilon_1 = \epsilon_5$. Substituting, we see that $\epsilon' = O(\epsilon^{556} \xi(\epsilon^{192})^2)$. (*End of Proof of Theorem* 3.7.6) $\blacksquare$

3.7.4 Extensions to Higher Order Norms

In this section, we show how to generalize the work from the previous section. Here we work with tensors of higher rank, mostly d and $(d+1)$. Suppose $f : \mathbb{F}_2^n \to \{-1, 1\}$ is a degree $d+1$ function. Given $Y = \langle y_1, \cdots, y_{d+1} \rangle \in (\mathbb{F}_2^n)^{d+1}$, by $\Gamma \cdot Y$ (or $Y \cdot \Gamma$) we will mean

$$\Gamma \cdot Y \stackrel{\text{def}}{=} \sum_{(i_1, \cdots, i_{d+1}) \in [n]^{d+1}} \Gamma_{i_1, \cdots, i_{d+1}} \prod_{j \in [d+1]} y_{j, i_j}.$$

Moreover note that for $Y = \langle y_1, \cdots, y_d \rangle \in (\mathbb{F}_2^n)^d$, $Y \cdot \Gamma$ (or $\Gamma \cdot Y$) is well defined and denotes a vector. Define $X \stackrel{\text{def}}{=} \langle x_1, \cdots, x_d \rangle \in (\mathbb{F}_2^n)^d$. We use $\mathbb{E}_X \hat{f}^2_X(X \cdot \Gamma)$ to denote $\mathbb{E}_{x_1, \cdots, x_d} \hat{f}^2_{x_1 \cdots x_d}(X \cdot \Gamma)$. If f is a degree $(d+1)$ polynomial, then it can be shown that there exists a tensor Γ of rank $d+1$ such that

$$\mathbb{E}_X \hat{f}^2_X(X \cdot \Gamma) = 1,$$

with Γ is invariant under the action of symmetric group Sym_{d+1}, where the group acts by permuting the indices of Γ. Moreover, it holds that for all $i_1, \cdots, i_d \in [n]$, $\Gamma_{i_1, i_1, i_2, \cdots, i_d} = 0$.

We now show a somewhat converse to the above.

Lemma 3.7.16 *Let Γ be a tensor of rank $d+1$ and that it is invariant under the action of Sym_{d+1} group. Further assume $\Gamma_{i_1,i_1,i_2,\cdots,i_d} = 0$ for all $i_1, \cdots, i_d \in [n]$. If*

$$\mathbb{E}_X \hat{f}_X^2(X \cdot \Gamma) \geq \epsilon_1,$$

Then there exists a d-degree polynomial h such that

$$\|f - h\| \leq \frac{1}{2} - \epsilon'.$$

Proof: As in the proof of Lemma 3.7.7, we first construct M in an analogous manner i.e., $\forall (i_1, \cdots, i_{d+1})$ such that $i_j = i_k \implies j = k$, we set

$$\sum_{\sigma \in \mathrm{Sym}_{d+1}} \sigma(\mathsf{M}_{i_1,\cdots,i_{d+1}}) = \Gamma_{i_1,\cdots,i_{d+1}},$$

and $\Gamma_{i_1,i_1,i_2,\cdots,i_d} = \sigma(\Gamma_{i_1,i_1,i_2,\cdots,i_d}) = 0$ for all $i_1, \cdots, i_d \in [n]$ and all $\sigma \in \mathrm{Sym}_{d+1}$. Denote $X' = \langle \underbrace{x, \cdots, x}_{d+1} \rangle$. Then define $g(x)$ so that $Łg(x) = \mathsf{M} \cdot X'$. Clearly then $Łg_X(x) = \langle x, \Gamma \cdot X \rangle$, up to a phase. Therefore,

$$\mathbb{E}_X \left[\mathbb{E}_z \, f_X(z) g_X(z) \right]^2 = \mathbb{E}_X \hat{f}_X^2(X \cdot \Gamma) \geq \epsilon_1$$

However, Lemma A.2.9 yields

$$\mathbb{E}_X \left[\mathbb{E}_z \, f_X(z) g_X(z) \right]^2 = \sum_\alpha \widehat{fg}_\alpha^4 \leq \max_\alpha \widehat{fg}_\alpha^2$$

Let β be a vector such that $|\widehat{fg}_\beta| \geq \sqrt{\epsilon_1}$. This then implies that there is a choice of $c \in \{0, 1\}$ such that for a degree $(d+1)$ polynomial $Łh(x) = X' \cdot \mathsf{M} + \langle x, \beta \rangle + c$ it holds that

$$\|f - h\| \leq \frac{1}{2} - \epsilon'.$$

■

Therefore, it suffices to show that $\|f\|_{U^{d+2}} \geq \epsilon \implies \exists \Gamma$ of rank $d+1$ with nice properties such that $\mathbb{E}_X \hat{f}_X^2(\Gamma \cdot X) \geq \epsilon_1$.

We now generalize the definition of vertical parallelogram to arbitrary di-

mension. We give a recursive definition.

Definition 3.7.17 *In $\mathbb{F}_2^n$, call a pair of elements $\{y, y+h\}$ a parallelogram of dimension 1 of size h at y. Given parallelograms P_1, P_2 of dimension d ($d \geq 1$) of size $(h_2, \cdots, h_{d+1})$ at $(x_2, \cdots, x_{d+1})$ and at $(x'_2, \cdots, x'_{d+1})$, we define a parallelogram of dimension $d+1$ of size $(h_1, \cdots, h_{d+1})$ at $(x_1, \cdots, x_{d+1})$ to be the set of points $\{x_1\} \times P_1 \cup \{x_1 + h_1\} \times P_2$. Henceforth, $\mathcal{P}$ will denote the set of all parallelograms of dimension d.*

Definition 3.7.18 *A function $f : (\mathbb{F}_2^n)^d \to \mathbb{F}_2^n$, is said to be d-linear if $\forall i \in [d], \forall (x_1, \cdots, x_d) \in (F_2^n)^d, y_i \in \mathbb{F}_2^n$ it holds that*

$$f(x_1, \cdots, x_d) + f(x_1, \cdots, x_{i-1}, y_i, x_{i+1}, \cdots, x_d) = f(x_1, \cdots, x_{i-1}, x_i + y_i, x_{i+1}, \cdots, x_d).$$

Given a function $f : (\mathbb{F}_2^n)^d \to \mathbb{F}_2^n$, we extend it on the set of parallelograms of dimension d in the natural way, i.e., if P is a parallelogram then $f(P) \stackrel{\text{def}}{=} \sum_{p_i \in P} f(p_i)$.

Lemma 3.7.19 *(Exact Characterization) A function $f : (\mathbb{F}_2^n)^d \to \mathbb{F}_2^n$ is d-linear iff $\forall P \in \mathcal{P}$ it holds that $f(h_1, \cdots, h_d) = f(P)$, where P is of size $(h_1, \cdots, h_d)$.*

Proof: Easy. ■

Conjecture 3.7.20 *For $d \geq 3$, let $\psi : (\mathbb{F}_2^n)^d \to \mathbb{F}_2^n$ be a function and, $A \subseteq (\mathbb{F}_2^n)^d$ be a set such that $|A| = \Omega(2^{dn})$. Let $P_{h_1, \cdots, h_d}$ denote parallelogram of length $(h_1, \cdots, h_d)$. It is given that*

$$\Pr_{(h_1, \cdots, h_d), P_{h_1, \cdots, h_d}} [\psi(h_1, \cdots, h_d) = \psi(P_{h_1, \cdots, h_d}) \mid \{(h_1, \cdots, h_2)\} \cup P_{h_1, \cdots, h_d} \subseteq A] \geq \epsilon.$$

Then there exists a refinement A' of A, i.e., $A' \subseteq A$ and $|A'| = \Omega(|A|)$, and a d-linear function $\theta : (\mathbb{F}_2^n)^d \to \mathbb{F}_2^n$ such that $\psi|_{A'} = \theta|_{A'}$ i.e., for all $p \in A'$, $\psi(p) = \theta(p)$. We set $|A'| = \xi(\epsilon)|A|$.

We now generalize Theorem 3.7.6.

Theorem 3.7.21 *Let $f : \{0,1\}^n \to \{-1, 1\}$ be a function such that $\|f\|_{U_{d+2}} \geq \epsilon$. Assuming Conjecture 3.7.20 holds, there exists a $d+1$ degree polynomial g such that the distance between f and g is at most $\frac{1}{2} - \epsilon'$, where $\epsilon' = \text{poly}\left(\epsilon^{2^{O(d)}}, \xi\left(\epsilon^{2^{O(d)}}\right)\right)$.*

We now define a product distribution on (symmetric) functions $\phi : (\mathbb{F}_2^n)^d \to \mathbb{F}_2^n$ by defining $\Pr[\phi(x_1, \cdots, x_d) = \alpha] = \hat{f}^2_{x_1 \cdots x_d}(\alpha)$. (Note that $\phi(x_1, \cdots, x_d) = \phi(x_{\sigma(1)}, \cdots, x_{\sigma(d)})$ holds for all $\sigma \in \mathrm{Sym}_d$.) Further the choices for distinct d-tuples are independent. Let $\rho = \Omega(1)$ be chosen later. For a point $p = (x_1, \cdots, x_d)$, we write $f_p(\cdot)$ to mean $f_{x_1 \cdots x_d}(\cdot)$ (and same as $f_X(\cdot)$).

Then define the random variable L on this probability space as follows:

$$L(\phi) \stackrel{\text{def}}{=} \Pr_{(h_1,\cdots,h_d)\in[n]^d, P\in\mathcal{P}} \Big[\phi(h_1, \cdots, h_d) = \phi(P); \hat{f}^2_{h_1 \cdots h_d}(\phi(h_1, \cdots, h_d)) \geq \rho;$$

$$\forall p \in P \;\; \hat{f}^2_p(\phi(p)) \geq \rho \Big].$$

Lemma 3.7.22 *If $\|f\|_{U^{d+2}} \geq \epsilon$, then*

$$\mathbb{E}_\phi(L(\phi)) \geq \epsilon_5$$

Proof: For a point $p = (x_1, \cdots, x_d) \in P$, we write $f_p(x)$ to denote $f_{x_1 \cdots x_d}(x)$. Also we naturally (implicitly) associate a mapping from the set of points of P to $[2^d]$. Given P a parallelogram in dimension d of length $(h_1, \cdots, h_d)$, one can view it as a collection of 1-dimensional parallelograms $\{z_i, z_i + h_d\}$ of length h_d indexed by a $(d-1)$-dimensional parallelogram Q of length $(h_1, \cdots, h_{d-1})$.

$$\mathbb{E}_\phi(L(\phi)) = \mathbb{E}_{h_1 \cdots h_d, P} \sum_{\alpha_1, \cdots, \alpha_{2^d}; \hat{f}^2_p(\alpha_p) \geq \rho; \hat{f}^2_{h_1 \cdots h_d}(\sum_i \alpha_i) \geq \rho} \hat{f}^2_{h_1 \cdots h_d}(\sum_i \alpha_i) \cdot \prod_{p \in P} \hat{f}^2_p(\alpha_p)$$

$$\geq \mathbb{E}_{h_1 \cdots h_d, P} \sum_{\alpha_1, \cdots, \alpha_{2^d}} \hat{f}^2_{h_1 \cdots h_d}(\sum_i \alpha_i) \cdot \prod_{p \in P} \hat{f}^2_p(\alpha_p) \; - \; (2^d + 1)\rho$$

$$= \mathbb{E}_{h_1 \cdots h_d, P, s} \; f_{h_1 \cdots h_d} * f_{h_1 \cdots h_d}(s) \cdot \prod_{p \in P} f_p * f_p(s) \; - \; (2^d + 1)\rho$$

$$= \mathbb{E}_{h_1 \cdots h_d, s, Q, z_1, \cdots, z_{2^{d-1}}} \; f_{h_1 \cdots h_{d-1} s} * f_{h_1 \cdots h_{d-1} s}(h_d) \cdot \prod_{p \in Q} f_{ps} * f_{ps}(z_i) f_{ps} * f_{ps}(z_i + h_d) - (2^d + 1)\rho$$

$$= \mathbb{E}_{h_1 \cdots h_d, s, Q, z_1, \cdots, z_{2^{d-1}}} \sum_{\{\alpha_{p,1}, \alpha_{p,2}\}_{p \in Q}, \beta} \hat{f}^2_{h_1 \cdots h_{d-1} s}(\beta) \chi_\beta(h_d) \prod_{p \in Q} \hat{f}^2_{ps}(\alpha_{p,1}) \chi_{\alpha_{p,1}}(z_i) \hat{f}^2_{ps}(\alpha_{p,2}) \chi_{\alpha_{p,2}}(z_i + h_d)$$

$$= \mathbb{E}_{h_1 \cdots h_{d-1}, s, Q} \sum_{\{\alpha_{p,1}, \alpha_{p,2}\}_{p \in Q}, \beta} \prod_{p \in Q} \delta^{\alpha_{p,1}}_{\alpha_{p,2}} \, \delta^{\sum_p \alpha_{p,1} + \alpha_{p,2}}_{\beta} \; \hat{f}^2_{h_1 \cdots h_{d-1} s}(\beta) \cdot \prod_{p \in Q} \hat{f}^2_{ps}(\alpha_{p,1}) \hat{f}^2_{ps}(\alpha_{p,2})$$

$$= \mathbb{E}_{h_1 \cdots h_{d-1}, s, Q} \sum_{\{\alpha_p\}_{p \in Q}} \hat{f}^2_{h_1 \cdots h_{d-1} s}(\sum_p \alpha_p) \cdot \prod_{p \in Q} \hat{f}^4_{sp}(\alpha_p)$$

Observe that

$$\|f\|^{2^{d+2}}_{U^{d+2}} = \mathbb{E}_u \|f_u\|^{2^{d+1}}_{U^{d+1}}.$$

Denote $\epsilon_2 = \frac{\epsilon^{2^{d+2}}}{2}$ and $\epsilon_3 = \epsilon_2^{\frac{1}{2^{d+1}}}$. Since $\|f\|_{U^{d+2}} \geq \epsilon$, we get by averaging

$$\Pr_u [\, \|f_u\|_{U^{d+1}} \geq \epsilon_3 \,] \geq \epsilon_2.$$

Call a u *good* if $\|f_u\|_{U^{d+1}} \geq \epsilon_3$. Let $\mathcal{F} = \{f_u \mid u \text{ is } \textit{good}\}$. Recall Equations 3.38, 3.39, 3.40. Following that we use induction to conclude that for any $g \in \mathcal{F}$, we have

$$\mathbb{E}_{h_1 \cdots h_{d-1}, Q} \sum_{\{\alpha_p\}_{p \in Q}} \prod_{p \in Q} \widehat{g}^2_p(\alpha_p) \cdot \hat{g}^2_{h_1 \cdots h_{d-1}}(\sum_p \alpha_p) \geq \frac{\epsilon_3^{2^{O(d)}}}{2^{O(d)}}.$$

This implies

$$\mathbb{E}_{h_1 \cdots h_{d-1}, s, Q} \sum_{\{\alpha_p\}_{p \in Q}} \prod_{p \in Q} \widehat{f}^2_{sp}(\alpha_p) \cdot \hat{f}^2_{s h_1 \cdots h_{d-1}}(\sum_p \alpha_p) \geq \epsilon_4 \stackrel{\text{def}}{=} \epsilon_2 \cdot \frac{\epsilon_3^{2^{O(d)}}}{2^{O(d)}}.$$

Lemma A.2.8 and some averaging argument now yields

$$\mathbb{E}_\phi(L(\phi)) \geq \epsilon_5 \stackrel{\text{def}}{=} \frac{\epsilon_4}{2} \cdot \Big(\frac{\epsilon_4}{2^{d+2}}\Big)^{2^{d+1}+1}. \qquad \blacksquare$$

Fix ϕ such that $L(\phi) \geq \epsilon_5$. For this function, define set

$$A \stackrel{\text{def}}{=} \{X = (x_1, \cdots, x_d) | \hat{f}^2_X(\phi(X)) \geq \rho\}.$$

Clearly $|A| \geq \sqrt{\frac{\epsilon_5}{2}}$.

An application of Conjecture 3.7.20 yields that there exists a tensor D of rank $(d+1)$, appropriately defined, such that

$$\mathbb{E}_X \hat{f}^2_X(\mathsf{D} \cdot X) \geq \xi(\epsilon_5) \cdot \rho \cdot \sqrt{\frac{\epsilon_5}{2}} \stackrel{\text{def}}{=} \epsilon_6$$

Further note that D is symmetric, i.e., for all $\sigma \in \mathrm{Sym}_d$, $\mathsf{D} \cdot X = \mathsf{D} \cdot \sigma(X)$. Denote $D \stackrel{\text{def}}{=} \mathsf{D}(x_1, \cdots, x_{d-1})$. Note that D is a matrix.

We now define a set of functions, one for each $(d-1)$-tuples $X' \stackrel{\text{def}}{=} \langle x_1, \cdots, x_{d-1} \rangle \in (\mathbb{F}_2^n)^{d-1}$, $g^{X'}(x_d) \stackrel{\text{def}}{=} (-1)^{\langle \sum_{i \in [d]} x_i, Dx_d \rangle}$ i.e., $\text{Ł}g^{X'} = \langle \sum_i x_i, \mathsf{D} \cdot X \rangle$. Then note that following Lemma A.2.10 we have

$$\forall x_1 \cdots x_{d-1} \qquad \mathbb{E}_{x_d}\, g^{X'}(x_d) \hat{f}_X^2(\mathsf{D} \cdot X) = \mathbb{E}_{x_d}\, \hat{f}_X^2(\mathsf{D} \cdot X).$$

Moreover, defining $F^{X'}(y) = \hat{f}^2_{X'y}(Dy)$, and using Claim A.2.4 we get

$$\forall X' \;\; \mathbb{E}_{x_d} g^{X'}(x_d) \hat{f}_X^2(\mathsf{D} \cdot X) = \mathbb{E}_{x_d} g^{X'}(x_d) F^{X'}(x_d)$$

$$= \sum_z \widehat{g^{X'}}(z) \widehat{F^{X'}}(z) = \sum_z \widehat{g^{X'}}(z) \mathbb{E}_{x_d} \hat{f}_X^2(\mathsf{D} \cdot X + z).$$

Arguing as in the preceding section, it can be shown that

$$\mathbb{E}_X\, \delta_{D^t x_d}^{D x_d}\, \hat{f}_X(\mathsf{D} \cdot X) \geq \epsilon_6^2.$$

Now for $X' \in \{0,1\}^{n(d-1)}$ define $S_{X'} = \{u | Du = D^t u\}$. Set $S = \cup_{X'} \{X'\} \times S_{X'}$. Note that $(x_1, \cdots, x_{d-1}, u) \in S$ implies $(x_1, \cdots, x_{d-2}, u, x_{d-1}) \in S$ (and all its Sym_d permutations). So we define Λ by first defining on the S by letting $\Lambda \cdot X \stackrel{\text{def}}{=} \mathsf{D} \cdot X$. This can further be extended to the whole space maintaining this property.

Observe that $\forall x_1, \cdots, x_{d+1}$ and for any permutation $\sigma \in \text{Sym}_{d+1}$ it holds that $\langle x_{d+1}, X \cdot \Lambda \rangle = \langle x_{\sigma(d+1)}, \sigma(X) \cdot \Lambda \rangle$. Thus we obtain

$$\mathbb{E}_X \hat{f}_X^2(\Lambda \cdot X) \geq \epsilon_7 \stackrel{\text{def}}{=} \epsilon_6^2,$$

where Λ is invariant under the action of Sym_{d+1}. Thus all we need to show that $\forall i, i_1, \cdots, i_{d-1}$, $\Lambda_{i,i,i_1,\cdots,i_{d-1}} = 0$. First define a set of functions, one for each $X' \in \{0,1\}^{n(d-1)}$, $h^{X'}(y) \stackrel{\text{def}}{=} (-1)^{\langle y, X' \cdot \Lambda y \rangle}$. Let T be the tensor of rank d defined[27] $\langle y, \mathsf{T} \cdot X' \rangle = \langle y, X' \cdot \Lambda y \rangle$. Define the following set of functions, one for each X', $h^{X'}(y) \stackrel{\text{def}}{=} (-1)^{\langle y, X' \cdot \Lambda y \rangle}$. Thus

$$\text{Ł}h^{X'}(y) = \langle y, X' \cdot \mathsf{T} \rangle.$$

We record the following corollary of Lemma 3.7.13.

Corollary 3.7.23 *Let f be a boolean function. For all $x_1, \cdots, x_{d-1}$, $\langle y, z \rangle = 1 \implies$*

[27] Even simply $\mathsf{T}_{i_1 \cdots i_d} = \Lambda_{i_1 i_1 \cdots i_d}$.

$\hat{f}_{x_1 \cdots x_d y}(z) = 0$.

Denote $X^i \stackrel{\text{def}}{=} \{x_1, \cdots, x_d\} \setminus \{x_i\}$ for $i \in [d]$. Then we have

$$\mathbb{E}_X \prod_{i \in [d]} h^{X^i}(x_i) \hat{f}_X^2(X \cdot \Lambda) = \mathbb{E}_X \hat{f}_X^2(X \cdot \Lambda).$$

Therefore, we have

$$\epsilon_7 \leq \mathbb{E}_X \prod_{i \in [d]} h^{X^i}(x_i) \hat{f}_X^2(X \cdot \Lambda) = 2^{-nd} \sum_{X, \forall i \, \langle x_i, \mathsf{T} \cdot X^i \rangle = 0} \hat{f}_X^2(X \cdot \Lambda)$$

Let $S \stackrel{\text{def}}{=} \{X \mid \langle x_i, \mathsf{T} \cdot X^i \rangle = 0\}$. Further denote $Y_{X^i} = \{x_i | (X^i, x_i) \in S\}$ (with the ordering enforced). Clearly Y_{X^i} is a linear space. Let $\{(e_{i_1}, \cdots, e_{i_d}) \mid i_1, \cdots, i_d \in [n]\} \subseteq (\mathbb{F}_2^n)^d$ be a basis. Define Γ on the basis point as follows. For any $E \stackrel{\text{def}}{=} (e_{i_1}, \cdots, e_{i_d})$ set $E \cdot \Gamma \stackrel{\text{def}}{=} \Gamma(e_{i_1}, \cdots, e_{i_d})$ where

$$\Gamma \cdot E \stackrel{\text{def}}{=} \begin{cases} 0 & \text{if } \exists j_1 \neq j_2 \text{ such that } i_{j_1} = i_{j_2}. \\ E \cdot \Lambda & \text{if } E \in S \\ z \text{ such that } \langle z, e_{i_1} \rangle = \cdots = \langle z, e_{i_d} \rangle = 0 \text{ and} \\ \langle z, e_k \rangle = \langle e_k, E \cdot \Lambda \rangle \; \forall k \in [n] \setminus \{i_1, \cdots, i_d\}, & \text{otherwise} \end{cases}$$

Now extend Γ d-linearly everywhere in $(\mathbb{F}_2^n)^d$. Clearly Γ is symmetric. Moreover, for all X, it holds that $\langle x_i, X \cdot \Gamma \rangle = 0$. To see this observe

$$\langle x_i, X \cdot \Gamma \rangle = \sum_{j_1, \cdots, j_{d+1}} x_{i,j_1} \prod_{k \in [d]} x_{k,j_{k+1}} \langle e_{j_1}, (e_{j_2}, \cdots, e_{j_{d+1}}) \cdot \Gamma \rangle$$

$$= \sum_{j_1, \cdots, j_{d+1}; j_1 \neq j_{i+1}} x_{i,j_1} \prod_{k \in [d]} x_{k,j_{k+1}} \big(\langle e_{j_1}, (e_{j_2}, \cdots, e_{j_{d+1}}) \cdot \Gamma \rangle$$

$$+ \langle e_{j_{i+1}}, (e_{j_2}, \cdots, e_{j_i}, e_{j_1}, e_{j_i+2}, \cdots, e_{j_{d+1}}) \cdot \Gamma \rangle \big) = 0.$$

Furthermore observe that for each i, $S = \cup_{x_i} \{X^i\} \times Y_{X^i}$. Therefore, for all

$X \in S$ it holds that $X \cdot \Gamma = X \cdot \Lambda$ and hence,

$$\mathbb{E}_X \hat{f}_X^2(X \cdot \Gamma) \geq \epsilon_7.$$

Proposition 3.7.24 *Γ is invariant under the action of Sym_{d+1} and also, $\Gamma_{i_1,i_1,\cdots,i_d} = 0$. Further it holds that*

$$\mathbb{E}_X \hat{f}_X^2(X \cdot \Gamma) \geq \epsilon_7.$$

Proof: Similar to the proof of Proposition 3.7.15. ■

This completes the proof of the theorem with the setting of $\epsilon_1 = \epsilon_7$. Substituting, we see that $\epsilon' = O\left(\epsilon^{2^{O(d)}} \xi \left(\epsilon^{2^{O(d)}}\right)^2\right)$. (*End of Proof of Theorem* 3.7.6) ■

3.8 Conclusion

We resolved the question posed in [AKK+03] for all prime fields. Independently in [KR04] the question has been resolved for all fields. The lower bound in Corollary 3.6.2 implies that our upper bound is almost tight.

In spite of long line of research in this area [RS96, FS95, BSSVW03, BSS03, BSHR03, EGH+04, Din06, Gol05], many questions still remain unanswered. We mention a few open problems.

- Characterization of LTCs : It is still not clear whether a given code is LTC. In that direction the lower bound given in [AKNS99] implies that if the dual distance is not constant, then the code is *not* locally testable with a constant number of queries.

  In Alon et. al. [AKK+03] the following conjecture has been proposed.

  Conjecture 3.8.1 *Any code with small (constant) dual distance, having a doubly transitive group acting on the coordinates of the codewords mapping the dual code to itself, is locally testable.*

  Interestingly several proofs demonstrating various natural codes[28] to be LTCs (eg., [RS96, FS95, AKK+03, JPRZ04, KR04]) ([KL05] being the notable exception) follow the self-correction based approach introduced in [RS96]. This

[28] We exclude here codes arising from PCPs.

approach critically uses the presence of the doubly transitive group to make the tests robust. Kaufman and Litsyn take a different approach, more combinatorial in nature, in [KL05]. Using the weight-distribution of the BCH code, they show that the dual of the BCH codes with designed distance $2t+1$ can be tested with $O(t/\epsilon)$ queries. They also give a sufficient condition for a code to be locally testable. The condition roughly says that if the number of fixed length codewords in the dual to the union of the code and its ϵ-far coset is suitably smaller than the same in the dual of the code, then the code is locally testable. Their argument is more combinatorial in nature and needs the knowledge of weight-distribution of the code and thus differs from the self-correction approach. This therefore does not allow (local) self-correction which is desired in many applications. They further define a notion of regular locally testable codes (which are testable by sampling constant weight codewords from its dual) that covers most known locally testable codes. Moreover they show that regular local testability of a linear code implies that the dual code is spanned by low-weight words, and therefore the dual has small (constant) distance. Interestingly, constant dual distance alone is known to be not sufficient for local testability (see [BSHR03]).

- In [GS02, BSSVW03], initiated by [GS02], the existence of good LTCs (i.e., almost constant rate and linearly growing distance) has been studied. In [BSS03], it has been shown that good cyclic locally testable codes do not exist. In [BSS05] an explicit LTC of rate nearly linear (inverse polylog) is given with polylog query complexity. Later in [Din06] the query complexity has been improved to a constant. Both of these constructions are based on PCPs of proximity. Is there a natural code that achieves this parameter? Also in [Gol05] it has been conjectured that no LTCs with constant rate exists.

It will be of tremendous interest to establish inverse theorems unconditionally, as this will enhance our understanding of low-degree polynomials considerably. An even more daunting task would be to generalize the results to (bounded) functions from finite Abelian groups to fields. Also, it is not clear for order d, what is the best $\xi(\cdot)$ that one can hope for.

Chapter 4

List Decoding over Bounded Alphabets

We define a new family of error-correcting codes based on algebraic curves over finite fields, and develop efficient list decoding algorithms for them. Our codes extend the class of algebraic-geometric (AG) codes via a generalization of the approach in the recent breakthrough work of Parvaresh and Vardy [PV05]. We begin with a formal definition of list-decoding.

Definition 4.0.2 *A code $\mathcal{C} \subseteq \mathbb{F}_q^N$ is said to be (ρ, L) (combinatorially) list decodable if for any vector $v \in \mathbb{F}_q^N$, the number of codewords that are within ρ-distance is at most L. Further, if there exists an efficient list-decoding algorithm that outputs such a list when given any word of length N, the code is said to have a (ρ, L) efficient list-decoding algorithm.*

Our main interest lies in the case when L is at most poly(N). The trade-off between the rate R and the error-correction radius p is a central one governing list decoding. We remind the readers that traditional "unique decoding" algorithms can achieve an error-correction radius only $(1 - R)/2$.

4.0.1 Previous Work on List Decoding

A simple probabilistic argument shows that $(1 - \epsilon, O(1/\epsilon))$ list-decodable codes of rate $\Omega(\epsilon)$ exist. Moreover a counting argument also shows that $\Omega(\epsilon)$ is the best

rate possible for such codes (see [Gur01]). An explicit construction of efficiently decodable optimal family of list-decodable codes is still open.

The seminal paper of Sudan [Sud97] and later improved by Guruswami and Sudan [GS99] show that Reed-Solomon codes (and algebraic geometry codes, also see [SW98]) with rate ϵ^2 are efficiently list decodable, up to an error radius of $1-\epsilon$. In [CS03] a list-decoding algorithm is given that corrects a fraction of error $1-\epsilon$ and achieves a rate of $\Omega(\epsilon^\alpha)$ (for any constant $\alpha > 1$). However the error model considered in that paper is probabilistic (and synchronous) and the probability that the correction is achieved is at least $\Omega(R^{M/(M+1)}))$, where the probability is over the random choices of error assuming a Q-ary symmetric channel .

Recently Parvaresh and Vardy in [PV05] constructed a code which is a variant of RS codes and can be list decoded beyond the $1-\sqrt{R}$ radius for rates $R \leq 1/16$. Thus the PV-code achieves a rate $\Omega(\epsilon/(\log(1/\epsilon)))$ to correct errors up to $1-\epsilon$ fraction in the highly noise setting.

Further building on the work of Parvaresh and Vardy, Guruswami and Rudra [GR06] have constructed explicit codes (called folded Reed-Solomon codes) that achieve list decoding capacity with polynomial encoding/decoding complexity.

4.0.2 Algebraic-Geometric Codes : A Brief Introduction

Most of the notation and terminology we use is standard in the study of algebraic-geometric codes, and can be found in Stichtenoth's book [Sti93]. We briefly recap some key facts concerning algebraic function fields and algebraic-geometric codes that we need for our description. For the purpose of exposition, we do not attempt to be technically correct for things that we would not need. We follow [vL98, Sti93].

Let F be a function field over $\mathbb{F}_q$, denoted $F/\mathbb{F}_q$, i.e., a finite algebraic extension of the field $\mathbb{F}_q(x)$ of rational functions over $\mathbb{F}_q$. Viewed differently, let I be a prime ideal in $\mathbb{F}_q[x,y]$. Then the set Υ of zeros of I is called an affine variety[1]. The ring $\mathbb{F}_q[x,y]/I$ is called the coordinate ring of the variety Υ. Observe that the coordinate ring is an integral domain. The quotient field of $\mathbb{F}_q[x,y]/I$, denote $F(\Upsilon)$, is the function field associated with variety Υ. For example, consider the ideal $\Upsilon_1 : y^2 - x = 0$ in $\mathbb{F}_q[x,y]$. Then the coordinate ring corresponding to Υ_1 is expressions of the form $\alpha + \beta y$ where $\alpha, \beta \in \mathbb{F}_q[x]$ and y satisfies the equation

[1] We avoid technicalities here.

$y^2 = x$. That is, the function field of Υ_1 is an algebraic extension of $\mathbb{F}_q(x)$ of degree two.

We now define discrete valuation.

Definition 4.0.3 *A discrete valuation of $F/\mathbb{F}_q$ is a function $v : F \to \mathbb{Z} \cup \{\infty\}$ with the following properties:*

- $v(x) = \infty \iff x = 0$.
- $v(xy) = v(x) + v(y)$
- $v(x + y) \geq \min\{v(x), v(y)\}$ *for any* $x, y \in F$.
- *There exists an element* $z \in F$ *with* $v(z) = 1$.
- $v(\alpha) = 0$ *for any* $\alpha \in \mathbb{F}_q^*$.

A subring R of F is said to be a valuation ring if for every $z \in F$, either $z \in R$ or $z^{-1} \in R$. Each valuation ring is a *local ring*, i.e., it has a unique maximal ideal. A place (prime divisor) P of the function field $F/\mathbb{F}_q$ is the maximal ideal of some valuation ring. If $\mathcal{O}$ is a valuation ring of $F/\mathbb{F}_q$ and P its maximal ideal, then $\mathcal{O}$ is uniquely determined by P, namely

$$\mathcal{O} = \{z | z^{-1} \notin P\}.$$

The elements in O/P are known as units. To each place P there is a unique valuation, denoted by $v_P : F \to \mathbb{Z} \cup \{\infty\}$. A uniformizer (or a local parameter) of a place P is a function $f \in F$ such that $v_P(f) = 1$.

To give an example, we consider the rational function field, $\mathbb{F}_q(x)$. Roughly, this can be seen as the function field corresponding to curve $x = y$. (Actually, it is a function field corresponding to the projective line, $(x : y)$, i.e., not both x, y are zero.) Consider a point $P_1 = (1 : 1)$. We observe that

$$\mathcal{O}_{P_1} = \{\frac{p(x,y)}{q(x,y)} \mid p(x,y), q(x,y) \in \mathbb{F}_q[x,y],\ gcd(p(x,y), q(x,y)) = 1,\ q(P_1) \neq 0\}$$

is a local ring. Now consider the function $f = \frac{y-x}{x}$. Observe that $f(P_1) = 0$. Further observe that f vanishes at P_1 with order one[2]. Therefore, f is a uniformizer of the

[2]Though we have not defined the order explicitly which requires an explicit valuation, informally it has the same meaning as in analysis.

place P_1. In fact, the unique maximal ideal of $\mathcal{O}_{P_1}$ is (f), i.e., the ideal generated by f. Observe that each function in $\mathbb{F}_q[x, y]$ can be written as $u \cdot f^n$ where n is an integer and u is an unit, i.e., $u \in O_{P_1}/(f)$. For example, consider the function $(x^2 - y^2)/y^2$. Observe that $(x^2 - y^2)/y^2 = -\frac{(y-x)}{x} \cdot \frac{(y+x)x}{y^2} = u_1 \cdot f$ where $u_1 = -x(x + y)/y^2$ is a unit.

The set of places of F will be denoted $\mathbb{P}_F$. Geometrically, this corresponds to the set of all non-singular[3] points on the algebraic curve corresponding to F. The picture here is that on a non-singular point P, we can define a local ring in the manner stated in the above example which comprises of rational functions that do not have a pole at P, i.e., either vanishes at P or is a unit. The maximal ideal at P consists of functions that vanishes at P. (See for details [Har77].)

The valuation ring corresponding to a place P is called the ring of regular functions at P and is denoted $\mathcal{O}_P$. Associated with a place P is a valuation $v_P : F \to \mathbb{Z}$, that measures the order of zeroes or poles of a function at P (with the convention $v_P(0) = \infty$). In terms of v_P, we have $\mathcal{O}_P = \{x \in F \mid v_P(x) \geq 0\}$ and $P = \{x \in F \mid v_P(x) > 0\}$. The quotient $\mathcal{O}_P/P$ is a field since P is a maximal ideal – it is called the residue field at P. The residue field $\mathcal{O}_P/P$ is a finite extension field of $\mathbb{F}_q$; the degree of this extension is called the *degree* of P, and is denoted $\deg(P)$. Informally, observe that a point is said to belong to a curve whenever the coordinates of the point satisfy certain equation. Now, it is totally possible that the coordinates of a point may lie on an extension field of $\mathbb{F}_q$. Loosely speaking, the order of this extension field is the degree of the point.

For every place P, we have an evaluation map $\mathsf{ev}_P : \mathcal{O}_P \to \mathcal{O}_P/P$ defined by $\mathsf{ev}_P(z) = z + P$; this map is $\mathbb{F}_q$-linear. We will think of ev_P as a map into $\mathbb{F}_{q^{\deg(P)}}$ using an isomorphism of the residue field to $\mathbb{F}_{q^{\deg(P)}}$. Alternatively, elements of F can be viewed as functions on $\mathbb{P}_F$ (hence the name *function field* for F): The evaluation of $z \in F$ and $P \in \mathbb{P}_F$, denoted $z(P) = \mathsf{ev}_P(Z)$, is either ∞ (if $z \notin \mathcal{O}_P$), or belongs to $\mathbb{F}_{q^{\deg(P)}}$.

Example 4.0.4 *Consider $F = \mathbb{F}_q(x)$, a rational function field. It is known*[4] *that*

[3]We do not define them here as we only consider curves that are smooth, i.e., do not have non-singular points. Interested readers can see [Har77].

[4]See [Sti93], Section 1.2

the places corresponding to F are

$$\mathbb{P}_F = \{p(x) \mid p \text{ is a monic irreducible poly } \} \cup \{\frac{1}{x}\}.$$

What happens to other functions, say eg., $(x-2)(x-3)$? Observe that this function behaves as a uniformizer at 2 where the $(x-3)$ part behaves as a unit, and as a uniformizer at 3 where the $(x-2)$ part behaves as a unit. Therefore, one cannot define a local ring at $(x-3)(x-2)$ as neither $(x-2)/(x-3)$ nor its inverse can be included in the local ring.

We will denote $\mathbb{P}_\infty = (1/x)$. Also, it is known that if $p \in \mathbb{P}_F$, then $\deg(p) =$ *degree(p), i.e., the traditional degree of the polynomial p. For example,* $\deg(x-\alpha) = 1$ *for some $\alpha \in \mathbb{F}$. Also,* $\deg(\mathbb{P}_\infty) = 1$. *Let $p(x) = t(x)/r(x)$ be a rational polynomial with $r(x)$ not identically zero. To compute $v_{\mathbb{P}_\infty}(p)$, observe that if degree $(r) >$ degree (t), then $p(x)$ vanishes at $\mathbb{P}_\infty$ of order exactly degree $(r)-$ degree(t). On the other hand, if degree $(r) <$degree(t), then $p(x)$ has a pole at $\mathbb{P}_\infty$ of order exactly degree $(t)-$ degree(r).*

With this observation, then note that

$$\mathcal{L}(K\mathbb{P}_\infty) = \{p(x) | v_{\mathbb{P}_\infty}(p) \geq -K, \text{, i.e., degree}(p) \leq K\}.$$

The set of divisors D_F of a function field $F/\mathbb{F}_q$ is the (additively written) free abelian group generated by the places $\mathbb{P}_F$. For a divisor $D = \sum_{P \in \mathbb{P}_F} n_P P$ where all but finitely many n_P are 0, its degree, denoted $\deg(D)$, is defined as $\deg(D) = \sum_{P \in \mathbb{P}_F} n_P \deg(P)$ (note that this is a finite sum). For a divisor $D = \sum_P n_P P$, we define the set of functions

$$\mathcal{L}(D) \stackrel{\text{def}}{=} \{z \in F \mid \forall P \in \mathbb{P}_F \; v_P(z) \geq -n_P\};$$

this forms a vector space over $\mathbb{F}_q$. To illustrate, we give an example.

We will not attempt to define the genus, which we denote by g, here. It is known to be always a non-negative integer and an intrinsic quantity of an algebraic curve. The readers should accept it as a by-product of the Riemann-Roch theorem (see [Sti93]). Further, it is known that curves with arbitrary high genus exist and can be efficiently constructed. For our purpose, we need a weaker version of Riemann-Roch which we state now.

Theorem 4.0.5 *(Weak Riemann-Roch) If $D \in D_F$ is a divisor of $F/\mathbb{F}_q$ of sufficiently large degree (i.e., at least $2g-1$), then* $\dim(\mathcal{L}(D)) = \deg(D) - g + 1$.

For a divisor $D = \sum_P n_P P$, define

$$\text{supp}(G) \stackrel{\text{def}}{=} \{P \mid n_P \neq 0\}.$$

An algebraic-geometric code over $\mathbb{F}_q$ is obtained by evaluating a carefully chosen subset of elements of F at distinct *places of degree one*, say $P_1, \cdots, P_N$ (these places can be treated as points on a curve), a divisor G such that $\text{supp}(G) \cap \{P_1, \cdots, P_N\} = \emptyset$. The geometric Goppa code $C_{\mathcal{L}}(D, G)$ associated with the divisors D and G is defined by

$$C_{\mathcal{L}}(D, G) \stackrel{\text{def}}{=} \{\langle x(P_1), \cdots, x(P_N)\rangle \mid x \in \mathcal{L}(G)\}.$$

Notice the necessity of having $\text{supp}(G) \cap \{P_1, \cdots, P_N\} = \emptyset$. This implies that no function in $\mathcal{L}(G)$ has a pole at the places $P_1, \cdots, P_N$, and hence, the code is well-defined.

Example 4.0.6 *(Reed-Solomon code) Let $\mathbb{F} = \mathbb{F}_q$ and[5] $F = \mathbb{F}(x)$. For each $\alpha \in \mathbb{F}_q$, define the place $P_\alpha = (x - \alpha)$. Further define $P_\infty = (1/x)$. Let F be a positive integer and set $G = (K-1)P_\infty$. Observe that the set $\mathcal{L}((K-1)P_\infty)$ consists of all those $z \in F$ for which z has no poles at places other than P_∞, and has less than K poles at P_∞. Using Theorem 4.0.5 we deduce that* $\dim(\mathcal{L}(G)) = K - 1 + 0 + 1 = K$, *which it should be. As said before,*

$$\mathcal{L}(G) = \{p| \ \textit{degree}(p) \leq K - 1\},$$

and hence, the code

$$C_{\mathcal{L}}(D, G) = \{\langle p(x-\alpha_1), \cdots, p(x-\alpha_N)\rangle = \langle p(\alpha_1), \cdots, p(\alpha_N)\rangle \mid \textit{degree } (p) \leq K-1\}.$$

Also, note that the minimum distance is at least $N - K + 1$ since a nonzero function in $\mathcal{L}(G)$ can have at most $K-1$ zeroes. (Since the sum of the order of zeros and order of poles of a function should sum to zero.)

If $G = KP_\infty$ for some positive integer K, then the code is also known as a

[5]The genus of a rational function field is 0.

one point code. There exist efficient implementations for these types of code which have made these codes the most sought after.

4.1 Construction of Correlated AG Codes

4.1.1 Overview

As mentioned above, in this work we propose a generalization of the Parvaresh-Vardy coding scheme to AG codes. While fairly natural in hindsight, the generalization to AG codes is not immediate, since, as we describe below, the special structure of RS codes and the rational function field $\mathbb{F}_q(X)$ are used in a more than superficial way in [PV05].

The ability to view a low-degree polynomial over $\mathbb{F}_q$ (i.e., the function being evaluated) also as a field element from a larger field $\mathbb{F}$, and operating on it in the field $\mathbb{F}$ to get another related polynomial is crucial to the PV construction. Indeed, the decoding is performed by solving a system of polynomial equations over the field $\mathbb{F}$ whose solutions contain all possible codewords that must be output. For Reed-Solomon codes, there is a natural way to view polynomials as field elements, since polynomials of degree $< K$ are in one-to-one correspondence with elements of the extension field $\mathbb{F}_q[X]/(E(X)) \approx \mathbb{F}_{q^K}$ (where $E(X)$ is an irreducible polynomial of degree K over $\mathbb{F}_q$). In order to generalize this framework to AG codes, we need an injective homomorphism from the elements of the function field F that are evaluated to give the AG-encoding (i.e., the analog of low-degree polynomials for the RS case) to a suitable field $\mathbb{F}$. We achieve this by associating with an element f of the function field, the field element in an extension field $\mathbb{F}_{\mathcal{P}} \stackrel{\text{def}}{=} \mathcal{O}_{\mathcal{P}}/\mathcal{P}$ of $\mathbb{F}_q$ which is the evaluation $f(\mathcal{P})$ of f at a fixed place $\mathcal{P}$ of large enough degree. This evaluation is then used to obtain, from the message function f, a correlated function h such that $h(\mathcal{P})$ is a carefully chosen function of $f(\mathcal{P})$. For function fields of larger genus this evaluation map restricted to the message functions can be made bijective; however it could be expensive. We do not know how to efficiently compute the necessary information needed for this bijection. We overcome this in [GP07] by noticing that an injection actually suffices. That is, we show that a correlated function h with the desired evaluation $h(\mathcal{P})$ always exists in a slightly larger space compared to the message space to which f belongs.

The decoding algorithm follows the "interpolation and find roots" idea that is common to [Sud97, GS99, PV05]. However, another technical complication arises in the phase when the interpolated polynomial, say $\mathcal{Q}$, is mapped into a polynomial $\mathcal{T}$ with coefficients from $\mathbb{F}_{\mathcal{P}}$ by evaluating each of its coefficients at the place $\mathcal{P}$. Following [PV05], we seek to find roots in $\mathbb{F}_{\mathcal{P}}$ of N, and using the above-mentioned injection from messages into $\mathbb{F}_{\mathcal{P}}$, map these roots back to obtain the list of messages. It is crucial that in this step $\mathcal{T}$ is a nonzero polynomial when $\mathcal{Q}$ is. For the Reed-Solomon case, this is easy to achieve, since the coefficients of $\mathcal{Q}$, which are polynomials over $\mathbb{F}_q$ in one variable, come from a *principal ideal domain* (PID), i.e., a ring all of whose ideals are generated by a single element. Therefore, the only way $\mathcal{T}$ can be zero when $\mathcal{Q}$ is nonzero, is if all coefficients of $\mathcal{Q}$ are divisible by the generator of the ideal $\mathcal{T}$ (i.e., by a univariate polynomial $E(X)$). In this case we can divide $\mathcal{Q}$ by the appropriate power of $E(X)$ to get a lower-degree nonzero polynomial $\tilde{Q}$ which is not divisible by $E(X)$, and then work with it instead.

However, for general function fields, the ring $\mathcal{O}$ containing the coefficients of $\mathcal{Q}$ typically is not a PID. Therefore, even if all coefficients of $\mathcal{Q}$ vanish at $\mathcal{T}$, they may not share a common factor in $\mathcal{O}$ and the above approach for RS codes cannot be applied. In [GP07], we circumvent this issue in two ways, giving two different algorithms. Here we outline only the second of them.

In this approach, we do not impose additional restrictions on the coefficients of $\mathcal{Q}$ beyond the usual interpolation based algorithms. Instead, if all coefficients of $\mathcal{Q}$ vanish at $\mathcal{P}$, we multiply each of the coefficients of $\mathcal{Q}$ by a function ν^c where ν is a function with a pole of order 1 at $\mathcal{P}$ and no poles elsewhere (such a function must exist if the degree of $\mathcal{P}$ is large), and $c \geq 1$ is the minimum of the zero orders at $\mathcal{P}$ of the coefficients of $\mathcal{Q}$. We then reduce the resulting polynomial $\mathcal{Q}' = \nu^c \mathcal{Q}$ modulo $\mathcal{P}$ to get a nonzero polynomial $\mathcal{T}$ with coefficients in $\mathbb{F}_{\mathcal{P}}$ and then proceed as in [PV05].

Several challenges arise in implementing this idea. In [GP07], we handle them as follows. First, we need a way to represent ν and a way to compute c. Also, the coefficients of $\mathcal{Q}'$ are no longer in the ring $\mathcal{O}$, making it difficult to represent and evaluate them efficiently. Nevertheless, we prove that the coefficients of $\mathcal{Q}'$ belong to a linear space of functions with bounded number of poles at $\mathcal{P}$. We use this to compute c as well as a representation of the coefficients of $\mathcal{Q}'$ that lets us evaluate them at $\mathcal{P}$ (assuming some extra preprocessed information). Here we assume that

the function ν is given explicitly and that c can be computed by multiplying proper powers of ν and checking whether the product vanishes.

The advantage of this approach is that we can use large multiplicities in the interpolation phase and as a result there is no degradation in error-correction radius compared to the results of Parvaresh-Vardy (for example, using two correlated functions already suffices to go beyond regular AG codes). The drawback is that the decoding algorithm needs more complicated, albeit still a polynomial amount of preprocessed information, and we do not know how to perform the preprocessing in polynomial time (but given the preprocessed information, the algorithm runs in polynomial time).

4.1.2 An Encoding Scheme

We now describe a correlated AG code construction where we use a **pair** of functions in the evaluation. The extension of the code, decoding algorithm, and analysis for the case when more than two correlated functions are evaluated as part of encoding, follows in a natural way, and will be discussed only briefly.

We now describe our construction of the code. Let F be a function field over $\mathbb{F}_q$ corresponding to a smooth, irreducible curve. Let g be the genus of F. Suppose F has at least $N+1$ places of degree one, say $x_0, x_1, \ldots, x_N$. Let $K \geq g$ be arbitrary (this assumption is mainly for convenience). We will describe a code C of block length N over alphabet $\mathbb{F}_{q^2}$ with q^K codewords. The rate of the code will thus be $\mathcal{R}(C) = K/(2N)$. The code will **not** be linear.

The messages of C will be identified with the vector space $\mathbb{F}_q^K$. We specify the code by specifying its encoding function, E, which will be an injective map $\mathsf{E} : \mathbb{F}_q^K \to \left(\mathbb{F}_{q^2}\right)^N$.

For our construction we need to carefully pick the divisor G. We now state what we need for our construction. Let $\mathcal{P}$ be a prime divisor not in the support of G and such that

$$\deg(\mathcal{P}) \;=\; \dim(\mathcal{L}(G)).$$

Assume $x_0, x_1, \cdots, x_n$ are disjoint from G and $\mathcal{P}$. The next lemma due to Felipe Voloch shows that we can choose $G, \mathcal{P}$ and $x_0, x_1, \cdots, x_n$ as conditioned.

Lemma 4.1.1 *Let $F/\mathbb{F}_q$ be a function field of genus $g \geq 2$ with $N_1 > 2g-2$ places of degree one. Given $a > 2g-2, a \leq N_1$, there exists a divisor G of degree a, a*

prime divisor $\mathcal{P}$ *of degree* $a+1-g$ *and rational places* $\{x_i\}_{i=0}^{n}$*, with* $n = N_1 - a - 1$ *such that* x_i *and* $\mathcal{P}$ *are not in the support of* G *and* $\dim \mathcal{L}(G - \mathcal{P}) = 0$*.*

Proof: Since $N_1 \geq 2g - 1 \geq g + 1$ we can apply ([BLB05], Proposition 4 (2)) to get a divisor D of F of degree $g - 1$ satisfying $\dim \mathcal{L}(D) = 0$. Choose any place $\mathcal{P}$ of degree $a - g + 1$. Further let G be a positive divisor linearly equivalent to $\mathcal{P} + D$, which exists since $\deg(\mathcal{P} + D) = a > 2g - 2$. Therefore, $G - \mathcal{P}$ is linearly equivalent to D. This implies $\dim \mathcal{L}(G - \mathcal{P}) = 0$. Furthermore since $G > 0$, G has at most $\deg G = a$ places of degree one in its support. Therefore there are at least $n + 1 = N_1 - a$ places of degree one not in the support of G. This completes the proof. ■

Though the above lemma proves the existence, we do not know how to compute G and $\mathcal{P}$ efficiently. From now on, we fix $x_0, x_1, \cdots, x_n, G, \mathcal{P}$ as ensured by Lemma 4.1.1. Denote the discrete valuation ring of $\mathcal{P}$ by $\mathcal{O}_{\mathcal{P}}$ and the residue field of $\mathcal{P}$ by $\mathbb{F}_{\mathcal{P}} \stackrel{\text{def}}{=} \mathcal{O}_{\mathcal{P}}/\mathcal{P}$. Now consider the natural map $\theta : \mathcal{L}(G) \to \mathbb{F}_{\mathcal{P}}$ induced by the inclusion $\mathcal{L}(G) \subset \mathcal{O}_{\mathcal{P}}$ (or equivalently, $\mathcal{L}(G) \ni f \mapsto f(\mathcal{P})$). Thus the condition that θ is a bijection is equivalent to it being an injection, which is clearly equivalent to $\dim(\mathcal{L}(G - \mathcal{P})) = 0$.

Moreover, since $\mathcal{O}_{\mathcal{P}} \to \mathcal{O}_{\mathcal{P}}/\mathcal{P}$ is a homomorphism of rings, we have for all $f, g \in \mathcal{L}(G)$, that

$$\theta(f \cdot g) = \theta(f) \cdot \theta(g). \tag{4.1}$$

Note that the the product on the left is in the function field F, whereas the product on the right is in $\mathbb{F}_{\mathcal{P}}$.

Let d be a constant to be chosen appropriately later. We define the following map from $\Gamma : \mathcal{L}(G) \to \mathcal{L}(G)$ that factors as

$$\mathcal{L}(G) \xrightarrow{\theta} \mathcal{O}_{\mathcal{P}}/\mathcal{P} \xrightarrow{d} \mathcal{O}_{\mathcal{P}}/\mathcal{P} \xrightarrow{\theta^{-1}} \mathcal{L}(G),$$

where the map d is defined as follows: $d : u \mapsto u^d$. Clearly Γ is a bijection. $\Gamma(f)$ should be seen as a correlated copy of the function f.

Let $\mathbb{F}_{q^2} = \mathbb{F}_q[\beta]$. Then consider the following map $\mathsf{E} : \mathcal{L}(G) \to \mathbb{F}_{q^2}^n$ by the map

$$\mathcal{L}(G) \ni f \mapsto \langle f(x_i) + \beta\Gamma(f)(x_i)\rangle_{i=1}^{n}$$

This code can clearly be seen as a generalization of Parvaresh-Vardy code ([PV05])

where the underlying Reed-Solomon code has been replaced by an algebraic geometry code.

For later use we mention the following fact.

Fact 4.1.2 *Let P be a rational point on F, i.e., a place of degree one. Then For every $f, h \in F$ with $v_P(f) = v_P(h)$ then there exists $\alpha, \gamma \in \mathbb{F}_q^*$ such that*

$$v_P(\alpha f + \gamma h) > \max\{v_P(f), v_P(h)\}.$$

Proof: Note $v_P(f) = -v_P(f^{-1})$ where $f^{-1} \in F$ is the inverse of f. Note therefore that $v_P(f \cdot f^{-1}) = v_P(h \cdot f^{-1}) = 0$. Therefore $f \cdot f^{-1} = \alpha \in \kappa_P^* = \mathbb{F}_q^*$ and $h \cdot f^{-1} = \gamma \in \kappa_P^* = \mathbb{F}_q^*$. Therefore $\alpha^{-1} f \cdot f^{-1} - \gamma^{-1} g \cdot f^{-1} = 0$, i.e., $f^{-1}(\alpha^{-1} f - \gamma^{-1} g) = 0$. Denote $f^* = (\alpha^{-1} f - \gamma^{-1} g)$. Then

$$0 < v_P(f^{-1} f^*) = -v_P(f) + v_P(f^*) \implies v_P(f^*) > v_P(f).$$

Renaming appropriately then yields the result. ∎

We also need the following theorem (see [Sti91]) which is essentially a a variant of Chinese Remainder Theorem.

Theorem 4.1.3 *(Weak Approximation Theorem) Let $F/\mathbb{F}_q$ be a function field, $P_1, \cdots, P_N$ pairwise distinct places of $F/\mathbb{F}_q$, $f_1, \cdots, f_N \in F$ and $n_1, \cdots, n_N \in \mathbb{Z}$. Then there exists an element $f \in F$ such that*

$$\forall i \in [N] \quad v_{P_i}(f - f_i) = n_i.$$

Next we outline the list-decoding algorithm.

4.2 A Decoding Algorithm

We use techniques from [SW98, GS99, PV05] to analyze the correctness of our list-decoding algorithm. Given a vector $\langle v_1, \cdots, v_N \rangle \in \mathbb{F}_{q^2}^N$, we view each v_i as $\langle y_i, z_i \rangle$ where $y_i, z_i \in \mathbb{F}_q$. Define $b \stackrel{\text{def}}{=} \ell - g + 1$, where ℓ is a parameter to be chosen appropriately later. We then try to fit the data points $\{(x_i, y_i, z_i)\}_{i=1}^N$ by a polynomial $\mathcal{Q}[y, z] \in F[y, z]$. We want that for each (x_i, y_i, z_i),

1. We get $\mathcal{Q}[y_i, z_i](x_i) = 0$ with multiplicity r which will be fixed later.

2. We also make sure that the $\mathcal{Q}$ has a small pole sum (over the support of G and x_0) ℓ, for any substitution of y, z with functions in $\mathcal{L}(G)$. Note that any function f in $L(G)$ satisfies $\sum_{x \in \text{supp}(G)} v_x(f) \geq -a$. We then want

$$\sum_{x \in \text{supp}(G) \cup \{x_0\}} v_x \left(\mathcal{Q} \left[y \leftarrow \mathcal{L}(G), z \leftarrow \mathcal{L}(G) \right] \right) \geq -\ell.$$

We will assume that we have been explicitly given functions $\psi_1, \cdots, \psi_b$, where $b \stackrel{\text{def}}{=} \ell - g + 1$, such that

- $\forall j \in [b]\ v_{x_0}(\psi_j) \geq 1 - g - j$ and
- $\forall j \in [b-1]\ v_{x_0}(\psi_j) > v_{x_0}(\psi_{j+1})$.

Clearly ψ_b's are linearly independent. We mention here that any set of $\{\psi_i\}_{i=1}^b$ satisfying the above will suffice for our case. We choose the set as follows. Let b^* be the smallest integer such that $\forall j \in [b], \psi_j \in \mathcal{L}(b^* x_0)$.[6] Since $\psi_1, \cdots, \psi_b$ are independent and belongs to $\mathcal{L}(b^* x_0)$, they form a basis of the vector space $\mathcal{L}(b^* x_0)$. We then define a divisor $H \stackrel{\text{def}}{=} b^* x_0 + G$. Note that by choice of x_0, G, we have $\text{supp}(H) \cap \{x_1, \cdots, x_N\} = \emptyset$. Also clearly $\mathcal{L}(G) \subseteq \mathcal{L}(H)$ and $\mathcal{L}(b^* x_0) \subseteq \mathcal{L}(H)$.

We set $s \stackrel{\text{def}}{=} \lfloor \frac{\ell - g}{a} \rfloor$. We then try to interpolate the following polynomial

$$\mathcal{Q}[y, z] = \sum_{j_2 + j_3 \leq s;\ j_2, j_3 \geq 0} \ \sum_{j_1 = 1}^{b - a(j_2 + j_3)} q_{j_1, j_2, j_3} \psi_{j_1} y^{j_2} z^{j_3}.$$

Note that

$$\begin{aligned}
\sum_{x \in \text{supp}(H)} v_x(\mathcal{Q}[y \leftarrow \mathcal{L}(G), z \leftarrow \mathcal{L}(G)]) &\geq \\
\sum_{x \in \text{supp}(H)} (v_x(\psi_{j_1}) + j_2 v_x(y \leftarrow \mathcal{L}(G)) + j_3 v_x(z \leftarrow \mathcal{L}(G))) &\geq \\
v_{x_0}(\psi_{j_1}) + \sum_{x \in \text{supp}(G)} (j_2 \cdot v_x(y \leftarrow \mathcal{L}(G)) + j_3 \cdot v_x(z \leftarrow \mathcal{L}(G))) &\geq
\end{aligned}$$

$$1 - g - j_1 - a j_2 - a j_3 \geq 1 - g - (b - a j_2 - a j_3) - a j_2 - a j_3 = 1 - g - b \geq -\ell,$$

[6] Essentially $b^* = \ell$ since we will later see $b > 2g$. However we do not need this fact.

this then automatically satisfies condition (2). (Since $v_{x_i}(\cdot)$ is an non-Archimedean valuation, it holds that $v_{x_i}(\mu f + \nu g) \geq \min(v_{x_i}(f), v_{x_i}(g))$, see Definition 4.0.3.) To get constraint (1) above, we shift our basis. We now recall a lemma from [GS99]. For completeness we reproduce the proof here. It basically proves the existence of a linear transform on the functional space over $\mathbb{F}_q$.

Lemma 4.2.1 *Given functions $\psi_1, \cdots, \psi_\sigma \in \mathcal{L}(b^* x_0)$ of distinct orders at x_0 satisfying $v_{x_0}(\psi_j) \geq 1 - g - j$ and a rational point $x_i \neq x_0$, there exists $\phi_1, \cdots, \phi_\sigma \in F^*$ with $v_{x_i}(\phi_j) \geq j - 1$ and such that there exists $\gamma_{x_i,j,h} \in \mathbb{F}_q$ where $1 \leq j, h \leq \sigma$ such that*

$$\psi_j = \sum_{h=1}^{\sigma} \gamma_{x_i,j,h} \phi_h. \tag{4.2}$$

Proof: As in [GS99], we prove a stronger statement by induction on σ: If $\psi_1, \cdots, \psi_\sigma$ are linearly independent (over $\mathbb{F}_q$) functions such that $v_{x_i}(\psi_j) \geq m$ for all $j \in [\sigma]$, then there are functions $\phi_1, \cdots, \phi_\sigma$ such that $v_{x_i}(\phi_j) \geq m + j - 1$ that generate the ψ_js over $\mathbb{F}_q$. Note that this then establishes the lemma.

W.l.o.g. assume the ψ_1 is a function with lowest order at x_i, by assumption $v_{x_i}(\psi_1) \geq m$. We then set $\phi_1 = \psi_1$. For $2 \leq j \leq \sigma$, we set $\psi'_j = \psi_j$ if $v_{x_i}(\psi_j) > v_{x_i}(\psi_1)$. If $v_{x_i}(\psi_j) = v_{x_i}(\psi_1)$, then by Fact 4.1.2, $\exists \alpha_j, \gamma_j \in \mathbb{F}_q^*$ such that the function $\psi'_j = \alpha_j \psi_1 + \gamma_j \psi_j$ satisfies $v_{x_i}(\psi'_j) > v_{x_i}(\psi_1) \geq m$. In this case, $\psi_j = \gamma_j^{-1} \psi'_j - \alpha_j \gamma_j^{-1} \psi_1$. This implies that $2 \leq j \leq \sigma$, $\phi_1 = \psi_1$ and ψ'_j generates ψ_j. Since $\psi'_2, \cdots, \psi'_\sigma$ are linearly independent and $v_{x_i}(\psi'_j) \geq m + 1$, by inductive hypothesis this yields $\phi_1, \cdots, \phi_\sigma$. ∎

With the above we can now express condition (1) on (x_i, y_i, z_i) being a zero of order at least r as satisfying a set of linear constraints. Note that shifting along x_i then yields

$$\mathcal{Q}[y, z] \;=\; \sum_{\substack{j_2 + j_3 \leq s; \\ j_2, j_3 \geq 0}} \; \sum_{j_1 = 1}^{b - a(j_2 + j_3)} \; \sum_{h=1}^{b} q_{j_1, j_2, j_3} \gamma_{x_i, j_1, h} \phi_h y^{j_2} z^{j_3}. \tag{4.3}$$

The shifting to y_i, z_i is achieved by defining $\mathcal{Q}^{(i)}[y, z](x) \stackrel{\text{def}}{=} \mathcal{Q}[y + y_i, z + z_i](x)$. Note that the terms in $\mathcal{Q}^{(i)}[y, z](x)$ that are divisible by $y^u z^v$ contribute $(u + v)$ towards the multiplicity of $(x_i, 0, 0)$ as a zero of $\mathcal{Q}^{(i)}$, or equivalently, the multiplicity of

(x_i, y_i, z_i) as a zero of $\mathcal{Q}$. Then

$$\mathcal{Q}^{(i)}[y,z](x) = \sum_{\substack{j_4+j_5\le s\\ j_4,j_5\ge 0}} \sum_{h=1}^{b} w_{h,j_4,j_5}\phi_h y^{j_4} z^{j_5}, \tag{4.4}$$

where

$$w_{h,j_4,j_5} \stackrel{\text{def}}{=} \sum_{j_2=j_4; j_3=j_5}^{j_2+j_3\le s} \sum_{j_1=1}^{b-a(j_2+j_3)} \binom{j_2}{j_4}\binom{j_3}{j_5} y_i^{j_2-j_4} z_i^{j_3-j_5} q_{j_1,j_2,j_3}\gamma_{x_i,j_1,h}. \tag{4.5}$$

Since $v_{x_i}(\phi_j) \ge j-1$, we achieve condition (2) on (x_i, y_i, z_i) being a zero of multiplicity r by imposing the following constraint

$$w_{h,j_4,j_5} = 0 \text{ for all } h \ge 1; j_4, j_5 \ge 0 \text{ such that } j_4 + j_5 + (h-1) \le r-1. \tag{4.6}$$

Therefore total number of constraints is $N \cdot r(r+1)(r+2)/6$. On the other hand the number of unknowns are $U \ge as(s+1)(s+2)/6$. In order to ensure that a non-zero $\mathcal{Q}[Y,Z]$ exists, we set

$$U = \frac{as(s+1)(s+2)}{6} \ge N \cdot \frac{r(r+1)(r+2)}{6} + 1. \tag{4.7}$$

Among the non-zero solutions of $\mathcal{Q}[Y,Z]$, we choose *the $\mathcal{Q}$ with the smallest* $v_{\mathcal{P}}(\mathcal{Q})$.

Lemma 4.2.2 *For $i \in [N]$, if $h \in F$ satisfies $h(x_i) = y_i$ and $\Gamma(h)(x_i) = z_i$, then $v_{x_i}(\mathcal{Q}[h, \Gamma(h)]) \ge r$.*

Proof: For such an i, $\mathcal{Q}[h,\Gamma(h)](x) = \mathcal{Q}^{(i)}[h(x) - h(x_i), \Gamma(h)(x) - \Gamma(h)(x_i)]$, i.e., localizing at x_i, and therefore,

$$\mathcal{Q}[h,\Gamma(h)](x) = \sum_{\substack{j_4+j_5\le s\\ j_4,j_5\ge 0}} \sum_{h=1}^{b} w_{h,j_4,j_5}\phi_h (h(x)-h(x_i))^{j_4} (\Gamma(h)(x) - \Gamma(h)(x_i))^{j_5}.$$

Since $w_{h,j_4,j_5} = 0$ for $j_4+j_5+(h-1) < r$, $v_{x_i}(\phi_h) \ge (h-1)$, and $v_{x_i}(h(x)-h(x_i))^{j_4} \ge j_4$ and $v_{x_i}(\Gamma(h)(x) - \Gamma(h)(x_i))^{j_5} \ge j_5$, we obtain

$$v_{x_i}(\mathcal{Q}[h,\Gamma(h)]) \ge r. \qquad \blacksquare$$

Lemma 4.2.3 *If $h \in \mathcal{L}(G)$ is such that $h(x_i) = y_i$ and $\Gamma(h)(x_i) = z_i$ for at least t values of $i \in [N]$, and $rt > \ell$, then $\mathcal{Q}[h, \Gamma(h)] \equiv 0$.*

Proof: By property 2 of the interpolated polynomial $\mathcal{Q}$, $\sum_{x \in \mathrm{supp}(H)} v_x \left(\mathcal{Q}[h, \Gamma(h)]\right) \geq -\ell$. Moreover clearly $\mathcal{Q}[h, \Gamma(h)] \in \mathcal{L}(H)$. However, at least on t points it holds that $h(x_i) = y_i$ and $\Gamma(h)(x_i) = z_i$. Therefore

$$\sum_{i=1}^{N} v_{x_i}(\mathcal{Q}[h, \Gamma(h)]) \geq r \cdot t > \ell.$$

Hence $\mathcal{Q}[h, \Gamma(h)]$ must be identically zero. ■

We next consider the image of $\mathcal{Q}[Y, Z]$ in $\mathbb{F}_{\mathcal{P}}[Y, Z]$ by evaluating at $\mathcal{P}$. We set $\mathcal{T}[Y, Z] = \mathcal{Q}[Y, Z](\mathcal{P})$. We show below that $\mathcal{T}[Y, Z]$ is not everywhere zero polynomial. Following this, we set $\mathcal{H}(Y) \stackrel{\text{def}}{=} \mathcal{T}[Y, Z \leftarrow Y^d]$ and find all the roots of $\mathcal{H}(Y)$. For each root ρ of $\mathcal{H}(Y)$, we output $\theta^{-1}(\rho)$.

We now prove that $\mathcal{T}[Y, Z]$ is not everywhere zero polynomial. Assume for contradiction that $\mathcal{Q}[Y, Z](\mathcal{P})$ is an everywhere zero polynomial. In the following lemma we then prove that there exists $\mathcal{Q}'[Y, Z]$ satisfying the interpolating conditions and the constraints such that when evaluated at $\mathcal{P}$ yields a non-zero polynomial. Furthermore, $v_{\mathcal{P}}(\mathcal{Q}'[Y, Z]) < v_{\mathcal{P}}(\mathcal{Q}[Y, Z])$. This contradicts our choice of $\mathcal{Q}[Y, Z]$, i.e., that $v_{\mathcal{P}}(\mathcal{Q}[Y, Z])$ is minimal.

Observer that

$$\mathcal{T}[Y, Z] = \mathcal{Q}[Y, Z](\mathcal{P}) \ = \sum_{\substack{j_2 + j_3 \leq s; \\ j_2, j_3 \geq 0}} \sum_{j_1 = 1}^{b - a(j_2 + j_3)} q_{j_1, j_2, j_3} \psi_{j_1}(\mathcal{P}) y^{j_2} z^{j_3}$$

is well defined since $\forall j \in [b]$ $v_{\mathcal{P}}(\psi_j) \geq 0$.

Lemma 4.2.4 *If $\mathcal{Q}[Y, Z]$ is not identically zero, then there always exists an $\mathcal{Q}'[Y, Z]$ such that it satisfies the conditions of being an interpolated polynomial and also satisfies all the linear constraints and Lemma 4.2.2 and Lemma 4.2.3, such that $\mathcal{T}[Y, Z] \stackrel{def}{=} \mathcal{Q}'[Y, Z]$ is not identically zero. Moreover it holds that $v_{\mathcal{P}}(\mathcal{Q}'[Y, Z]) < v_{\mathcal{P}}(\mathcal{Q}[Y, Z])$.*

Proof: Assume $\mathcal{Q}[Y, Z] \in F[Y, Z]$ is not identically zero. If $\mathcal{Q}[Y, Z](\mathcal{P})$ is not all-zero polynomial in $\mathbb{F}_{\mathcal{P}}[Y, Z]$ then we are done. Otherwise consider the image of

$\mathcal{Q}[Y,Z]$ in $\mathcal{O}_{\mathcal{P}}[Y,Z]$. Then clearly for any coefficient $\eta \in F$ of $\mathcal{Q}[Y,Z]$, $v_{\mathcal{P}}(\eta) \geq 1$. Let

$$\nu = \min\{v_{\mathcal{P}}(\eta_{ij}) \mid \mathcal{Q}[Y,Z] = \sum_{i,j} \eta_{ij} Y^i Z^j, \eta_{ij} \in F\}.$$

Clearly $\nu > 0$, otherwise $\mathcal{Q}[Y,Z](\mathcal{P})$ would not be identically zero polynomial. Now consider the function f such that

$$v_{\mathcal{P}}(f) \;=\; 1 \text{ and } \forall i \in \{0,1,\cdots,N\}\; v_{x_i}(f) \;=\; 0 \text{ and } \forall P \in \text{supp}(G)\; v_P(f) \;=\; 0.$$

By weak approximation theorem (Theorem 4.1.3) we know that such an f exists. Clearly f is a uniformizing parameter at $\mathcal{P}$.

Then consider $\mathcal{Q}'[Y,Z] \stackrel{\text{def}}{=} f^{-\nu} \cdot \mathcal{Q}[Y,Z]$, or equivalently, starting interpolation assuming a basis $f^{-\nu} \cdot \psi_1, \cdots, f^{-\nu} \cdot \psi_b$. First of all note that $\mathcal{T}[Y,Z](\mathcal{P}) \stackrel{\text{def}}{=} \mathcal{Q}'[Y,Z](\mathcal{P})$ is not all-zero polynomial. Also, it is clear that $v_{\mathcal{P}}(\mathcal{Q}'[Y,Z]) < v_{\mathcal{P}}(\mathcal{Q}[Y,Z])$. We need to show that $\mathcal{Q}'$ satisfies the condition as well as all the constraints.

Since f does not have any zero or pole at $\{x_i\}_{i=1}^N$, first condition is easily satisfied. Furthermore since f does not have zero or pole anywhere in the support of G and x_0, hence condition two is also satisfied.

Now we show that $\mathcal{Q}'[Y,Z]$ satisfies all the constraints. Note that Equation 4.2 can be reformulated as

$$f^{-\nu} \cdot \psi_j = \sum_{h=1}^{\sigma} \gamma_{x_i,j,h} f^{-\nu} \cdot \phi_h. \tag{4.8}$$

Hence Equation 4.3 now can be written as

$$\mathcal{Q}'^{(i)}[y,z](x) = \sum_{j_4+j_5 \leq s; j_4,j_5 \geq 0} \sum_{h=1}^{b} w_{h,j_4,j_5} f^{-\nu} \cdot \phi_h y^{j_4} z^{j_5}, \tag{4.9}$$

where w as in Equation 4.5. Also, recall $v_{x_i}(f) = 0$ for all $i = 1, \cdots, N$. Therefore Lemma 4.2.2 holds that is $\forall i \in [N] v_{x_i}(\mathcal{Q}'[h, \Gamma(h)]) \geq r$. Moreover since $\mathcal{Q}'[Y,Z] = f^{-\nu} \cdot \mathcal{Q}[Y,Z]$, we get $\mathcal{Q}'[h, \Gamma(h)] = 0$.

This completes the proof of the lemma. ■

W.l.o.g. the polynomial as ensured by Lemma 4.2.4 will be denoted by $\mathcal{Q}[Y,Z]$. Further we set $\mathcal{T}[Y,Z] = \mathcal{Q}[Y,Z](\mathcal{P})$. Note that $\mathcal{T}[Y,Z]$ is well-defined

and not everywhere zero polynomial. We now assume that $d \geq s+1$, where s is as defined earlier.

Lemma 4.2.5 *If $h \in \mathcal{L}(G)$ has agreement at least t, then $\mathcal{T}[\theta(h), (\theta(h))^d] = 0$.*

Proof: As mentioned before θ is an homomorphism, i.e., $\theta(f \cdot g) = \theta(f) \cdot \theta(g)$. Denote $h' = \Gamma(h)$. In that case note that $\mathcal{Q}[h, h'] \equiv 0$. Now,

$$\mathcal{Q}[h, h'] = \sum_{j_2+j_3 \leq s;\ j_2, j_3 \geq 0} \sum_{j_1=1}^{b-a(j_2+j_3)} q_{j_1,j_2,j_3} \psi_{j_1} h^{j_2} h'^{j_3} =$$

$$\sum_{\substack{j_2+j_3 \leq s; \\ j_2, j_3 \geq 0}} \left(\sum_{j_1=1}^{b-a(j_2+j_3)} q_{j_1,j_2,j_3} \psi_{j_1} \right) h^{j_2} h'^{j_3}$$

Therefore,

$$\sum_{j_2+j_3 \leq s;\ j_2, j_3 \geq 0} \left(\sum_{j_1=1}^{b-a(j_2+j_3)} q_{j_1,j_2,j_3} \psi_{j_1}(\mathcal{P}) \right) h^{j_2} h'^{j_3} = 0$$

Note that for all the non-zero q_{j_1,j_2,j_3}, $\psi_{j_1}(\mathcal{P})$ is well defined. Now applying θ, we get

$$\sum_{j_2+j_3 \leq s;\ j_2, j_3 \geq 0} \left(\sum_{j_1=1}^{b-a(j_2+j_3)} q_{j_1,j_2,j_3} \psi_{j_1}(\mathcal{P}) \right) (\theta(h))^{j_2} (\theta(h'))^{j_3} = 0$$

However, note that $\theta(h') = \theta(\Gamma(h)) = (\theta(h))^d$. Thus we get

$$\sum_{j_2+j_3 \leq s;\ j_2, j_3 \geq 0} \left(\sum_{j_1=1}^{b-a(j_2+j_3)} q_{j_1,j_2,j_3} \psi_{j_1}(\mathcal{P}) \right) (\theta(h))^{j_2} (\theta(h))^{dj_3} = 0$$

However, note that above is nothing but $\mathcal{T}[\theta(h), (\theta(h))^d]$. ■

Following the choice of d and an argument similar to [PV05] the next lemma is immediate (or see the proof of Lemma 4.2.10)

Lemma 4.2.6 *$\mathcal{H}(Y) = \mathcal{T}[Y, Z \leftarrow Y^d]$ is not identically zero.*

Therefore following Lemma 4.2.5 and Lemma 4.2.6, it is clear that whenever a codeword $C_{\mathcal{L}}(D, G)(h)$ has agreement at least t with the received word, then $\theta(h)$ appears as a root of $\mathcal{H}(Y)$. This completes the proof of correctness of the algorithm.

4.2.1 Choice of Parameters

Observe that so far what we have described holds true for any function field. For the best performance, we use the the function fields with the best possible ratio of g/n. Specifically, for q a square, we use a sequence of function fields with increasing genus for which g/n is at least $\frac{1}{\sqrt{q}-1}$ [TVZ82, GS95b].

Below we outline the parameters for this specific AG curves. We assume $t > \sqrt[3]{na^2}$.

1. $g \geq 2$ be the genus of $F/\mathbb{F}_q$.
2. $\mathcal{R} \stackrel{\text{def}}{=} K/(2N) = (a - g + 1)/(2N) \approx a/(2N)$
3. $s = \lfloor \frac{\ell - g}{a} \rfloor$
4. $b = \ell - g + 1 \geq as + 1$
5. $d \geq s + 1$
6. In order to ensure Equation 4.7 and $rt > \ell$, we set

$$r \stackrel{\text{def}}{=} \lfloor \frac{g + 1 + 2\sqrt[3]{Na^2}}{t - \sqrt[3]{Na^2}} \rfloor,$$

$$\ell \stackrel{\text{def}}{=} rt - 1$$

7. Note that the list size could be at most the degree of $\mathcal{H}$, i.e.,

$$L \leq degree(\mathcal{H}) \leq sd = s(s + 1) = \Theta(s^2).$$

8. Letting $\epsilon = (a/N)^{3/2}$, we get

$$\mathcal{R} + \frac{g}{2N} = \frac{a + 1}{2N} \approx \frac{\epsilon^{3/2} + 1}{2}.$$

Since our curve has $g/N \geq 1/(\sqrt{q} - 1)$, we get $q = \Omega(1/\epsilon^3)$.

Therefore this corrects up to $1 - t/N \approx (1 - \sqrt[3]{\frac{Na^2}{N^3}}) = (1 - (4R)^{2/3})$. We sum up everything in the following theorem.

Theorem 4.2.7 *For the choice of $d = s + 1$, the decoding algorithm described in Section 4.2 correctly finds all the codewords $c = \langle c_1, \cdots, c_N \rangle$ of E which satisfy $c_i = y_i + \beta z_i$ for at least t values of $i \in [N]$, for $t > \sqrt[3]{N(K+g-1)^2}$. Moreover, for some $c > 1$, if $t \geq c\sqrt[3]{N(K+g-1)^2}$, then the list output by the algorithm has size at most $O((\frac{c}{c-1})^2 (N/K)^{2/3})$. Further, there is a polynomial sized representation of the codes given which encoding and list decoding up to this radius can be performed in polynomial time.*

4.2.2 Extension to Multivariate case

We let $G, \mathcal{P}, H$ and $x_0, x_1, \cdots, x_N$ as before (as given in Lemma 4.1.1). Let $Q = q^M$, where M is an integer ≥ 2 and $\mathbb{F}_Q = \mathbb{F}_q[\beta]$. Also let $\{d_i\}_{i=1}^{M-1}$ are positive integers to be specified later. Corresponding to each d_i we consider a map $\Gamma_i : \mathcal{L}(G) \to \mathcal{L}(G)$ as follows:

$$\mathcal{L}(G) \xrightarrow{\theta} \mathcal{O}_{\mathcal{P}}/\mathcal{P} \xrightarrow{d_i} \mathcal{O}_{\mathcal{P}}/\mathcal{P} \xrightarrow{\theta^{-1}} \mathcal{L}(G),$$

where the map d_i is defined as $d_i : u \mapsto u^{d_i}$ and the map $\theta : \mathcal{L}(G) \to \mathbb{F}_{\mathcal{P}}$ is as before. We set[7]

$$d_0 \stackrel{\text{def}}{=} 1 \text{ and } d_i \stackrel{\text{def}}{=} d_{i-1}s + 1.$$

Then consider the following map $\mathsf{E} : \mathcal{L}(G) \to \mathbb{F}_Q^N$ by the map

$$\mathcal{L}(G) \ni f \mapsto \langle f(x_i) + \sum_{j=1}^{M-1} \beta^j \Gamma_j(f)(x_i) \rangle_{i=1}^N$$

This map is a natural generalization of the code considered in [PV05].

4.2.3 Multivariate Decoding

Given a vector $\langle v_1, \cdots, v_N \rangle \in \mathbb{F}_Q^n$, we view each v_i as $\langle z_{0,i}, z_{M-1,i} \rangle$ where $z_{j,i} \in \mathbb{F}_q$ for all $j = 0, \cdots, M-1$. We then try to fit the data points $\{(x_i, z_{0,i}, \cdots, z_{M-1,i})\}_{i=1}^N$ by a polynomial $Q[z_0, \cdots, z_{M-1}] \in F[z_0, \cdots, z_{M-1}]$. For convenience we write

[7]For ease of exposition, we will omit the fact that d_i needs to be co-prime to the order of $\kappa_{\mathcal{P}}^*$.

$z = \langle z_0, \cdots, z_{M-1} \rangle$. We interpolate the following polynomial with the following condition:

1. We get $\mathcal{Q}[z_0, \cdots, z_{M-1}](x_i) = 0$ with multiplicity r, that will be fixed later.

2. We also make sure that the $\mathcal{Q}$ has a small pole sum (over the support of H) ℓ, which will be fixed later, for any substitution of y, z with a function in $\mathcal{L}(G)$. Note that any function f in $\mathcal{L}(G)$ satisfies $\sum_{x \in \text{supp}(G)} v_x(f) \geq -a$. We want

$$\sum_{x \in \text{supp}(G) \cup \{x_0\}} v_x \left(\mathcal{Q} \left[\forall i \in \{0, \cdots, M-1\} z_i \leftarrow L(G) \right] \right) \geq -\ell.$$

With the definition of s, b, ψ_i from the Section 4.2, we then interpolate the following polynomial

$$\mathcal{Q}[z_0, \cdots, z_{M-1}] = \sum_{\substack{j_0 + j_1 + \cdots + j_{M-1} \leq s; \\ j_0, \cdots, j_{M-1} \geq 0}} \sum_{j=1}^{b - a(j_0 + j_1 + \cdots + j_{M-1})} q_{j, j_0, j_1, \cdots, j_{M-1}} \psi_j z_0^{j_0} \cdots z_{M-1}^{j_{M-1}}.$$

It is easy to see that condition 2 holds immediately. Also, by a similar argument as in Section 4.2 the condition 1 can be shown to hold provided we satisfy a few linear constraints as before. By a similar argument as in Section 4.2, we now estimate the number of variables in this system. We need the following claim:

Claim 4.2.8 *Let r and M be positive integers. Then it holds that*

$$\sum_{t=1}^{r} \frac{t(t+1)\cdots(t+M)}{(M+1)!} = \sum_{t=1}^{r} \binom{M+t}{t-1} = \binom{M+r+1}{r-1}$$

$$= \frac{r(r+1)\cdots(r+M)(r+M+1)}{(M+2)!}$$

Proof: Use induction on r. ∎

We now estimate the number of variables. Let U denote the total number of unknowns. Then clearly

$$\begin{aligned} U &= \sum_{j_0+\cdots+j_{M-1}\le s, j_i\ge 0} \sum_{j=1}^{b-a(j_0+\cdots+j_{M-1})} 1 \\ &= \sum_{j'=0}^{s} \sum_{j_0+\cdots+j_{M-1}=j'} \sum_{j=1}^{b-aj'} 1 \\ &= \sum_{j'=0}^{s} \sum_{j_0+\cdots+j_{M-1}=j'} (b-aj') \\ &= \sum_{j'=0}^{s} (b-aj') \binom{M+j'-1}{j'} \end{aligned}$$

$$\begin{aligned} &= \frac{1}{(M-1)!} \sum_{j'=0}^{s} (b-aj')(j'+1)(j'+2)\cdots(j'+M-1) \\ &\ge \frac{1}{(M-1)!} a \sum_{j'=0}^{s} (s-j')(j'+1)(j'+2)\cdots(j'+M-1) \\ &= as \sum_{j'=0}^{s} \frac{(j'+1)\cdots(j'+M-1)}{(M-1)!} - a \sum_{j'=0}^{s} \frac{j'(j'+1)(j'+2)\cdots(j'+M-1)}{(M-1)!} \\ &= as \sum_{j'=1}^{s+1} \frac{j'(j'+1)\cdots(j'+M-2)}{(M-1)!} - a \sum_{j'=1}^{s} \frac{j'(j'+1)(j'+2)\cdots(j'+M-1)}{(M-1)!} \\ &= a\frac{s(s+1)\cdots(s+M)}{M!} - aM\frac{s(s+1)\cdots(s+M)}{(M+1)!} \\ &= a\frac{s(s+1)\cdots(s+M)}{(M+1)!} \end{aligned}$$

Similarly we estimate the number of constraints which can be shown to be the N-times the number of distinct tuples such that

$w_{h,j'_0,\cdots,j'_{M-1}} = 0$ for all $h \ge 1; j'_0, \cdots, j'_{M-1} \ge 0$ such that $j'_0+\cdots+j'_{M-1}+(h-1) \le r-1$.

Lemma 4.2.9 *The number of distinct tuples such that*

$w_{h,j'_0,\cdots,j'_{M-1}} = 0$ *for all* $h \geq 1; j'_0, \cdots, j'_{M-1} \geq 0$ *such that* $j'_0 + \cdots + j'_{M-1} + (h-1) \leq r-1$,

is $\frac{r(r+1)\cdots(r+M)}{(M+1)!}$.

Proof: We induct on M. Clearly it holds for $M = 0$. Assume it holds for M. We now do the induction. Then number of tuples such that $\sum_{i=0}^{M} j_i + (h-1) \leq (r-1)$ is clearly

$$\begin{aligned}
\sum_{j_M=0}^{r-1} \# \left(\sum_{i=0}^{M-1} j_i + (h-1) \leq (r-1-j_M) \right) &= \sum_{j_M=0}^{r-1} \frac{(r-j_M)\cdots(r-j_M+M)}{(M+1)!} \\
&= \sum_{j_M=1}^{r} \frac{j_M \cdots (j_M+M)}{(M+1)!} \\
&= \frac{r \cdots (r+M+1)}{(M+2)!}
\end{aligned}$$

This completes the induction. ∎

Thus the total number of constraints is

$$N \cdot \frac{r(r+1)\cdots(r+M)}{(M+1)!}$$

Therefore in order to have a non-zero interpolating polynomial we require

$$a\frac{s(s+1)\cdots(s+M)}{(M+1)!} > N \cdot \frac{r(r+1)\cdots(r+M)}{(M+1)!} \tag{4.10}$$

Denote $Z = \langle z_0, \cdots, z_{M-1} \rangle$. Among the non-zero solutions of $\mathcal{Q}[Z]$, we choose the $\mathcal{Q}$ with the smallest $v_{\mathcal{P}}(\mathcal{Q})$. To list-decode we do as before. We consider the image of $\mathcal{Q}[Z]$ in $\mathbb{F}_{\mathcal{P}}[Z]$ by evaluating at $\mathcal{P}$, i.e., we set

$$\mathcal{T}[Z] \stackrel{\text{def}}{=} \mathcal{Q}[Z](\mathcal{P}).$$

Next we set $\mathcal{H}(Y) = \mathcal{T}[\forall i \in \{0, \cdots, M-1\}\ z_i \leftarrow Y^{d_i}]$ and factor $\mathcal{H}(Y)$. For each root ρ of $\mathcal{H}(Y)$ we count the agreement of the codeword $\mathsf{E}(\theta^{-1}(\rho))$ with the received word. If the agreement is at least t, then we output $\theta^{-1}(\rho)$.

Following an argument similar to Lemma 4.2.4, it can be shown that $\mathcal{T}[Z]$ is not identically zero. Further an argument similar to Lemma 4.2.5, it can be established that if there is a codeword in E corresponding to $h \in \mathcal{L}(G)$ that has agreement at least t with the received word, then

$$\mathcal{T}[z_0 \leftarrow h, z_1 \leftarrow \Gamma_1(h), \cdots, z_{M-1} \leftarrow \Gamma_{M-1}(h)] = 0.$$

To prove that $\mathcal{H}(Y) = \mathcal{T}[\forall i \in \{0, \cdots, M-1\}\ z_i \leftarrow Y^{d_i}]$ is not identically zero. We argue as in [PV05].

Lemma 4.2.10 *$\mathcal{H}(Y) = \mathcal{T}[\forall i \in \{0, \cdots, M-1\}\ z_i \leftarrow Y^{d_i}]$ is not identically zero.*

Proof: We consider the following M polynomials defined as follows:

$$\mathcal{H}_i(Y, z_{i+1}, \cdots, z_{M-1}) \stackrel{\text{def}}{=} \mathcal{H}_{i-1}(Y, Y^{d_i}, z_{i+1}, \cdots, z_{M-1}),$$

with $\mathcal{H}_0(Y, z_1, \cdots, z_{M-1})$. Clearly $\mathcal{H}_{M-1}(Y) = \mathcal{H}(Y)$. We prove by induction that none of the polynomials $\mathcal{H}_i$ is the all-zero polynomial. By our choice of d_1 and a similar argument as in Lemma 4.2.6, $\mathcal{H}_1(Y, z_2, \cdots, z_{M-1}) = \mathcal{H}_0(Y, Y^{d_1}, z_2, \cdots, z_{M-1})$ is non-zero. Assume as induction hypothesis that $\mathcal{H}_i$ is non-zero for some $i \geq 1$. We prove that $\mathcal{H}_{i+1}$ is non-zero as well.

$\mathcal{H}_{i+1}$ is a all-zero polynomial iff $z_i - Y^{d_i}$ is a factor of $\mathcal{H}_i$. However from the definition of $\mathcal{T}, Q$ and $\mathcal{H}_i(\cdot)$, it is clear that the no monomial in $\mathcal{H}_i$ has Y-degree larger than $s \cdot \max\{d_0, d_1, \cdots, d_{i-1}\}$. Thus the choice of d_i ensures that the polynomial $\mathcal{H}_{i+1}$ is not the all-zero polynomial. ■

4.2.4 Parameters

Below we give the parameters. We assume $t > \sqrt[M+1]{Na^M}$.

1. $g \geq 2$.
2. $\mathcal{R} \stackrel{\text{def}}{=} K/(MN) = (a - g + 1)/(MN) \approx a/(MN)$
3. Note that the alphabet size is $Q = q^M$.
4. $s = \lfloor \frac{\ell - g}{a} \rfloor$
5. $b = \ell - g + 1 \geq as + 1$

6. $d_0 \stackrel{\text{def}}{=} 1$ and $d_i \stackrel{\text{def}}{=} d_{i-1}s + 1$.

7. In order to ensure Equation 4.10 and $rt > \ell$, we set

$$r \stackrel{\text{def}}{=} \lfloor \frac{g + 1 + M\sqrt[M+1]{Na^M}}{t - \sqrt[M+1]{Na^M}} \rfloor,$$

$$\ell \stackrel{\text{def}}{=} rt - 1.$$

8. Note that the list size could be at most the degree of $\mathcal{H}$ i.e.,

$$L \leq degree(\mathcal{H}) \leq s \max\{d_0, \cdots, d_{M-1}\} \leq (s+1)^M = \Theta(s^M).$$

9. $Q = q^M$.

Theorem 4.2.11 *For small enough $\epsilon > 0$ and all integers $m \geq 2$, there is a family of Q-ary codes with $Q = O\left((1/m\epsilon)^{2(m+1)}\right)$ which has rate $\Omega(\frac{1}{m}(m\epsilon)^{(m+1)/m})$ and which is $(1-\epsilon, O(m\epsilon(m)^{2m/(m+1)}))$-list decodable. Furthermore, the codes have a representation, computable in expected polynomial time, that permits polynomial time encoding and polynomial time list decoding up to radius $(1-\epsilon)$ with polynomial sized preprocessed information.*

For decoding up to a fraction $(1-\epsilon)$ of errors, with the choice $m = \Theta(\log(1/\epsilon))$ in the above theorem, we get the following.

Corollary 4.2.12 *For all $\epsilon > 0$, there is a family of Q-ary codes with $Q = (1/\epsilon)^{O(\log(1/\epsilon))}$ which has rate $\Omega(\epsilon/\log(1/\epsilon))$ and which is $(1-\epsilon, (1/\epsilon)^{O(\log\log(1/\epsilon))})$-list decodable. Furthermore, the codes have a polynomial sized representation that permits encoding and list decoding up to radius $(1-\epsilon)$ in polynomial time.*

4.2.5 Complexity of Encoding/Decoding

We assume that we have precomputed $G, \mathcal{P}$ which we do not know how to compute efficiently. Further we do not know how to efficiently compute the function ensured by the weak approximation theorem. However, we can assume that this is given to us as a polynomial sized advice. The basis $\{\psi_j\}_{j=1}^b$ can be efficiently computed. Therefore all we need to do is to set up the linear system and solve for the roots of a polynomial in $\mathbb{F}_{\mathcal{P}}$.

We can avoid computing the divisor G by changing the code slightly (without changing the asymptotic performance of the code) as done in [GP07] (essentially due to Venkatesan Guruswami). The idea is that we do no longer insist on a bijection. Instead, we can embed $\mathcal{L}(G)$ in slightly larger space as such embedding can be shown to exist even when G is chosen to be simple, i.e., supported on a rational place.

In [GP07] one can find a randomized algorithm to compute the high degree place $\mathcal{P}$ for the specific "optimal" AG codes based on a tower of function fields due to Garcia and Stichtenoth [GS95b, GS96a]. The algorithm is due to Venkatesan Guruswami and runs in expected polynomial time (i.e., *Las Vegas*). Though not explicit in the sense of deterministic polynomial time constructibility, the representation is guaranteed to be correct and constructing it (a one time job) takes polynomial time with overwhelming probability. This level of explicitness should thus suffice for using the code. We remark that even for the algorithm of Guruswami and Sudan [GS99, GS01] (that achieved a decoding radius of at most $1 - \sqrt{\mathcal{R}}$), it was not known how to compute the required representation efficiently.

4.3 Extension to List Recovering and Binary Codes

4.3.1 List Recoverable Codes

Definition 4.3.1 *A code $C \subseteq \Sigma^N$ is* (γ, l, L)-list recoverable *if for every sequence of sets $S_1, S_2, \ldots, S_N$, where each $S_i \subseteq \Sigma$ has at most l elements, the number of codewords $c \in C$ which satisfy $c_i \in S_i$ for at least γN values of $i \in \{1, 2, \ldots, N\}$ is at most L.*

Note a code being (ρ, L)-list decodable is the same thing as it being $(1 - \rho, 1, L)$-list recoverable, so the above notion is more general than list decoding. The name list recovering was coined in [GI01] and this notion has played a crucial role in new constructions of list-decodable codes since. List recovering was first explicitly studied in work on extractor codes [TZ04]; the name was coined in [GI01], and it has played a crucial role in combinatorial constructions of list decodable codes, including those with linear complexity algorithms [GI03].

We now make the following observation. The algorithm in Section 4.2 can be trivially generalized to handle the case when there is a set S_i consisting of possibly more than one triple (y_i, z_{i1}, z_{i2}) for each location i. We simply need to add a

constraint for each such triple in the interpolation of Step 2, so that the total number of constraints will now be the total number of triples N (or in other words the total size of all the S_i's). It immediately follows that we get an algorithm for list recovering that works with agreement t with N replacing the block length n. Of course, a similar generalization also holds for the m-variate decoding algorithm and the agreement bound of.

Theorem 4.3.2 *For all integers $l \geq 2$, for all $\gamma > 0$ and all integers $m \geq 2$, there is a family of Q-ary codes for $Q = O((ml^{1/m}/\gamma)^{2(m+1)^2/m})$ which has rate $\Omega(\gamma/m^2 \cdot (\gamma/l)^{1/m})$ and which is (γ, l, L)-list recoverable for $L = O(m^2 \cdot m! \cdot (l/\gamma)^{(m+1)/m})$. Moreover, the codes have a natural representation that permits polynomial time encoding as well as polynomial time (γ, l, L)-list recovering with poly-sized pre-processed information.*

Corollary 4.3.3 *For all integers $l \geq 2$ and all $\gamma > 0$, there is a family of Q-ary codes for $Q = l^{O(\log\log(l/\gamma))} \cdot (1/\gamma)^{O(\log(1/\gamma))}$ which has rate $\Omega(\gamma/\log(l/\gamma))$ and which is (γ, l, L)-list recoverable for $L = (l/\gamma)^{O(\log\log(l/\gamma))}$. Moreover, the codes have a natural representation, that permits polynomial time encoding as well as polynomial time (γ, l, L)-list recovering with poly-sized pre-processed information.*

4.3.2 Binary Codes for List Decoding up to Radius $(1/2 - \epsilon)$

We now consider the problem of constructing binary codes for list decoding up to radius $(1/2-\epsilon)$, for small $\epsilon > 0$. Using our codes as the outer code in a concatenation scheme with a constant-sized binary inner code with Q codewords and rate $\Omega(\epsilon^2)$ and that is $(1/2 - \epsilon/2, l)$-list decodable, we can show the following.

Theorem 4.3.4 *For every $\epsilon > 0$, there is a family of binary codes of rate $\Omega(\epsilon^3/\log(1/\epsilon))$ that is $(1/2 - \epsilon, (1/\epsilon)^{O(\log\log(1/\epsilon))})$-list-decodable. The codes can be encoded and list-decoded in polynomial time for radius $(1/2-\epsilon)$ assuming a pre-processed information of polynomial size.*

We remark that the recent construction of [GR06] achieves a rate of $\Omega(\epsilon^3)$ for $(1/2 - \epsilon, L)$-list-decodable codes, but their construction time as well as list size L is $N^{\Omega(1/\epsilon^3)}$. In contrast, our codes are uniformly constructive, i.e., can be constructed and decoded in time $f(\epsilon)N^{O(1)}$ with exponent of n independent of ϵ, and achieve a list size independent of the block length.

4.4 Conclusion

Guruswami and Rudra have recently proposed an explicit list-decodable code that achieve the list decoding capacity. Their code is essentially a folded Reed-Solomon codes which is essentially Reed-Solomon code, but packed in a very clever way so that it looks like a PV-code. Informally, the key idea is that viewed appropriately, certain automorphisms of a rational function fields induce low degree map on bounded-degree polynomials. Formally, they prove the following theorem in [GR06].

Theorem 4.4.1 *([GR06], Theorem 2) Let p be a prime, $b > a \geq 1$ be two integers that are coprime to p, and m be an arbitrary integer. Further let $t \geq m^2 - m$ be an integer that is not a multiple of p, and even if $p > 2$. Let q be a power of p such that $q \equiv 1 \pmod{t \cdot a \cdot b}$. Let γ be a generator of $\mathbb{F}_q^*$. Define $k = a(q-1)/b$ and $e = \frac{(q^a-1)b}{a(q-1)}$. Then the following statements are true:*

1. *The polynomial $E(x) = x^k - \gamma$ is irreducible over $\mathbb{F}$.*

2. *For any polynomial $f(x) \in \mathbb{F}[x]$ of degree at most $k-1$ and $\ell \geq 1$,*

$$(f(x))^{q^{a\ell}} \pmod{E(x)} = f(\gamma^{e\ell}x).$$

Moreover, if $1 \leq \ell < m^2 - m$ then the above map is non-trivial, i.e., $\gamma^{e\ell} \neq 1$.

An open question is to generalize the above theorem over general function field. We strongly believe such generalization can yield even better list decodable codes than [GR06], that is achieving capacity over smaller alphabet and with much smaller list size (independent of n) and complexity.

Binary code with list decoding radius $1 - \epsilon$ and rate $\Omega(\epsilon^2)$ is known to be optimal. The explicit construction of this optimal family of polynomial time list decodable binary code is still open.

Chapter 5

Applications to Cryptography

In this Chapter, we present technique to lower bound the minimum distance of certain types of quasi-cyclic codes with large dimension by reducing the problem to lower bounding the minimum distance of a few significantly smaller dimensional codes. This analysis enables us to strengthen SHA-1 by changing the underlying code.

5.1 Overview

We start recalling the SHA-1 message expansion code, which is a binary linear code of dimension 512. The 512 information bits are viewed as sixteen 32-bit words $\langle W_0, \cdots, W_{15} \rangle$, and 64 additional words are generated by the recurrence:

$$W_i = (W_{i-3} \oplus W_{i-8} \oplus W_{i-14} \oplus W_{i-16}) \ <<< \ 1 \quad \text{for } i \ = \ 16, \cdots, 79, \qquad (5.1)$$

where the notation "<<<" denotes left rotation by one bit. The 80 words $\langle W_0, \cdots, W_{79} \rangle$ can be seen as constituting a code-word in a linear code over $\mathbb{F}_2$ with the above parity check equations. Unfortunately, this code has a minimum distance or weight of no more than 44. Further, the weight restricted to the last 60 words is only 25. This has been exploited in [WYY05c] to give a differential attack on SHA-1 with complexity 2^{69} hash operations. Recently, the complexity has further been improved to 2^{63} hash operations [WYY05a].

The code for SHA-0, which is the same as Equation 5.1 but without the rotation, has an even worse minimum weight. The small minimum weight of these codes

is an integral part of the attack strategies on these hash functions (see [Wan97a, Wan97b, CJ98, BC04b, BC04a, WYY05b, WYY05c]). The question naturally arises as to why codes with better minimum weight were not employed, even though the coding theory literature [vL98] is rife with codes with proven good minimum weight. The reason is that none of them comes close to being as efficient to implement in software as the code in Equation 5.1 above. One is then led to ask if codes more complex than above, but still easy to implement, could be shown to have a better minimum distance. Surprisingly, it was not even known how to lower bound the minimum weight of the above SHA-1 code, even though it is related to codes such as the Hadamard code [vL98] (we elaborate on this later).

The purpose of our work is three-fold. First, we introduce a new technique for lower bounding efficient-to-implement codes such as given by Equation 5.1. Second, we use this technique to lower bound this particular code (which was an open problem). Third, we show how one can design efficient-to-implement codes with a much better minimum distance, and to actually give such a code. We expect our technique to be helpful in designing future practical collision-resistant hash functions.

We recall the definition of quasicyclic codes which are natural generalization of cyclic codes. For more on quasicyclic codes, see [TW67, Che92, Lal03, LS05].

Definition 5.1.1 *An $[n, k]$ linear code $\mathcal{C}$ over $\mathbb{F}_q$ is a quasicyclic code if $\mathcal{C}$ is closed under cyclic shifts by ℓ places, for some $1 \leq \ell < n = \ell \cdot n'$, i.e.,*

$$\mathcal{C} \ni \langle c_1, \cdots, c_{n'\ell} \rangle \implies \langle c_{\ell+1}, \cdots, c_{n'\ell}, c_1, \cdots, c_\ell \rangle \in \mathcal{C}.$$

The smallest such ℓ is called the index of $\mathcal{C}$. Quasicyclic codes with index one, i.e., $\ell = 1$ are also known as cyclic codes in the literature.

We now elaborate on our technique. To do that, let us examine the specific code we analyze, as this specific example will help in understanding the complexity of the problem and the intricacy of the technique. The code we consider is an 80×32 length binary code of dimension 16×32, given by the following recurrence relation

(or parity check equations):

$$W_i = \begin{cases} W_{i-3} \oplus W_{i-8} \oplus W_{i-14} \oplus W_{i-16} \oplus ((W_{i-1} \oplus W_{i-2} \oplus W_{i-15}) \ <<< \ 1) \\ \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{if } 16 \leq i < 36 \\ W_{i-3} \oplus W_{i-8} \oplus W_{i-14} \oplus W_{i-16} \oplus ((W_{i-1} \oplus W_{i-2} \oplus W_{i-15} \oplus W_{i-20}) \ <<< \ 1) \\ \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{if } 36 \leq i \leq 79 \end{cases} \tag{5.2}$$

We will show that this code has minimum distance 82, and that moreover in just the last 64 words (contrast this with SHA-1 which has minimum weight at most 30 [WYY05c], and 192 for the highly inefficient Reed Solomon code described in Section 5.2). Of course, since the dimension of this code is 16×32, a brute force search of $2^{16\times 32}$ is infeasible. We will give a way to do a feasible search for this type of codes. Further, it is known that computing minimum weight of an arbitrary linear code is NP-hard (see [Var97]), and that approximating within a constant factor is NP-hard under randomized reduction (see [DMS03]). Observe that the code is a quasicyclic code with index 80. One can then hope to use the lower bound given in [Lal03], but that requires computing over $\mathbb{F}_{2^\ell}$ where ℓ denotes the index of the quasicyclic code. In our cases the index turns out to be quite large and such estimates are no longer computationally feasible.

We now briefly explain the main idea of our technique, using the above example code given by Equation 5.12. See that any codeword is represented as a 80×32 matrix. In the following, we prefer to call the rows as words. Now observe that either (a) there are no all-zero columns in the codeword, in which case we would like to show that on average there are a few (three, e.g.) non-zero bits in each column, or (b) there is a zero column in the codeword, in which case we would like to show that the code projected on a few columns, say $m << n$, has a large minimum distance.

Unfortunately, there are two major hurdles in this plan, related to case (b). Consider the first non-zero column next to a zero column (either to the left or the right). It turns out that the code projected on that column is not expected to be any better than the code for SHA-0, and hence we do not expect a minimum weight of more than 15-20 for that column. Thus, we would need m to be about five to get a minimum weight of 75, in which case the dimension of the projected code is still too

large, i.e., 16×5. Further, there are *pathological cases* (which cannot be avoided) where the code projected on a column yields a minimum weight as low as 1. Thus, we may be forced to consider m much larger than five. The novelty of our approach lies in tackling these two major hurdles. We show that the minimum weight of the sub-code in case (b) can be lower bounded by a function of the minimum weight of a few codes (some of which are subspaces), each of dimension at most 16×3. A "lazy" brute force search with early-stopping then yields a lower bound of 82.

5.2 Limitations of Purely Algebraic Techniques

We first investigate the SHA-0 code restricted to a single column, which is a length 80 binary code of dimension 16, given by the binary parity check equations:

$$a_i = a_{i-3} \oplus a_{i-8} \oplus a_{i-14} \oplus a_{i-16} \quad \text{for } i \ = \ 16, \cdots, 79 \tag{5.3}$$

The above parity check equation can be associated with the the polynomial $h(X) = X^{16} + X^{13} + X^8 + X^2 + 1$ over $\mathbb{F}_2$, which is known to be a primitive polynomial. Therefore, the smallest n such that $h(X)$ divides $X^n - 1$ is $2^{16} - 1$. Hence, if the above code was extended up to length $2^{16} - 1$, it would be the dual code of the cyclic code generated by the ideal $h(X)$. The following theorem of [KLP68] (Theorem 1) proves that the SHA-0 code is a truncated Hadamard code. Recall that the binary Hadamard code is the same as the (Generalized) Reed-Muller code of order one.

Theorem 5.2.1 *([KLP68]) Let G_ν denote the νth order punctured[1] GRM code over $\mathbb{F}_q$. The dual code C_d of G_ν is cyclic and α^r is a root of the generator polynomial $g_d(X)$ if and only if the weight of r is at least ν, where the weight of r is*

$$W(r) = \sum_{i=0}^{m-1} \delta_i \text{ and } r = \sum_{i=0}^{m-1} \delta_i q^i,$$

i.e., the expansion in radix-q form.

Note that the roots of the polynomial $h(X)$ in $\mathbb{F}_{2^{16}}$ are $\alpha, \alpha^2, \alpha^4, \cdots, \alpha^{2^{15}}$. Hence, by the above theorem, $h(X)$ has the same roots and degree as that of the generator

[1] See subsection 2.1.1 for the definition.

of the dual code of the punctured first order GRM. This proves that SHA-0 is a truncated Hadamard code.

Punctured Hadamard codes have an extremely good minimum distance of $2^{15} - 1$, or fractional distance 1/2. Unfortunately, we do now know of anything useful which can be said about this code truncated to just the first 80 bits, based purely on known algebraic methods. In fact, any such code (i.e., using any degree 16 primitive polynomial) has a minimum weight of at most 26, i.e., a fractional distance of less than 1/3 (as can be checked by a computer).

The lack of purely algebraic techniques to lower bound even this single column code emphasizes the difficulty of analyzing the more complex codes such as SHA-1 and that given by Equation (5.2). Of course, if $h(X)$ above was not primitive, and divided $X^{80} - 1$, then we would get a cyclic code of length 80. Such codes can be analyzed much more easily, and it is not too difficult to see that the best cyclic code gives a minimum distance of only 8. However, there are non-cyclic linear codes known of minimum distance 31, though they are really difficult to encode. One could also consider cyclic codes of length 85, which have a much better minimum distance and then truncate them. However, the analysis does not extend to codes which do column mixing like SHA-1.

Instead of quasi-cyclic codes as SHA-1 or Equation 5.2, one could consider cyclic codes of length 80×32, or of an appropriate length. First note that a random code will give minimum distance roughly 475 for a code with rate 1/4 and length 64×32 (follows from the Gilbert-Varshamov bound). Of course, finding such a code is infeasible. Alternatively, one can try a Reed Solomon code over $\mathbb{F}_{2^8}$ of length $2^8 - 1$ (bytes), and dimension 64 (bytes). Such a code has distance $256 - 64 = 192$ (over bytes). However, the encoder for this code requires multiplication by various elements in $\mathbb{F}_{2^8}$, and is not at all suitable for software implementations. A binary cyclic code of dimension 16×32 would also be extremely cumbersome to implement. Similar considerations rule out known good quasi-cyclic codes.

5.3 Intuition behind the Code

Let us start by examining why the message expansion code in SHA-1 given by Equation (5.1) is not satisfactory (observed independently in [RO05] and [MP05]).

We can rewrite Equation (5.1) as follows:

$$\forall i, 0 \leq i \leq 63, \quad W_i = W_{i+2} \oplus W_{i+8} \oplus W_{i+13} \oplus (W_{i+16} >>> 1), \tag{5.4}$$

where ">>> 1" denotes a one bit rotation to the right. The above clearly shows that a difference created in the last 16 words propagates to only up to 4 different bit positions. To see this, assume the 79^{th} word is all zero except at the zeroth position. Further assume that all the fifteen words starting from 64^{th} words till 78^{th} words are zero. We can do this as we can choose any consecutive 16 words as the message. Then observe that 79^{th} word can influence only 63^{rd} words setting its 1st column to 1. Now observe that 63^{rd} words can influence 61^{st} words, but it can only influence its 1st column. It can only cause a disturbance on the 2nd column only at the word $47th$. Therefore, we can for sure say that columns 5th onwards will be entirely zero.

One way to remedy this situation is to let $W_i = (W_{i+2} >>> 1) \oplus W_{i+8} \oplus W_{i+13} \oplus (W_{i+16} >>> 1)$. Now Equation (5.1) becomes $W_i = (W_{i-3} \oplus W_{i-8} \oplus W_{i-16}) <<< 1 \oplus W_{i-14}$. Thus, whether one considers the evaluation in the *forward direction* or in the *reverse direction*, the spread of differences to the neighboring columns (i.e., neighboring bits) is more frequent. However, it is not enough to just have a good intuition about the code, but one also needs to prove a good lower bound on the minimum weight of such codes.

The strategy we use to prove lower bounds on such codes is to divide the proof into two main cases. We argue that either there are no zero columns in a codeword and ensures a minimum average weight per column or starting from an all zero column, the first few neighboring non-zero columns are actually codewords in some good codes, in the sense that each of them has good minimum distance.

Elaborating on the first case, i.e., when there are no zero columns, if every column has weight at least three, we are done as the weight is then at least 96. So, assume that there is some non-zero column which has weight at most two. Thus, there are $(64 \times 63)/2 + 64$ choices for picking these bits in the column. Having picked these bits, the neighboring column is completely specified by at most 16 bits in that column (follows from the code equation). Now the two columns together have either weight 6, in which case we are maintaining an average of 3 per column, or the weight of these two columns is at most 5. Thus, as promised before our search is quite restricted. We continue in this fashion, noting that the code has to

be designed carefully so as to satisfy a property as in Claim 5.4.6.

As for the second case, we consider a contiguous band of zero columns, bordered on both sides with non-zero columns (we prove that they cannot be same; in fact we prove by a rank argument that there must be at least four consecutive non-zero columns). We have to assure that when a column is zero, and the neighboring column is non-zero (whether to the right or left), the resulting code for the neighboring column is a good code, i.e., with a good minimum weight. Note that this is important since we may possibly have at most 5-6 non-zero columns. Therefore it is desired that the disturbance propagates fast across columns. Unfortunately, this is impossible for the codes we are considering so far.

Consider a SHA-1 like code, with dimension 16×32, and which is invariant under column rotations. Moreover, suppose that the code is of the form

$$W_i = \sum_{t=1}^{16} a_t W_{i-t} + \left(\left(\sum_{t=1}^{16} b_t W_{i-t} \right) <<< 1 \right), \tag{5.5}$$

where $a_1, \cdots, a_{16}, b_1, \cdots, b_{16}$ are boolean. If a_{16} and b_{16} are equal, then there is a codeword which is zero everywhere, except for W_0 which is the all one 32-bit word. Thus for the sake of the argument, assume that $b_{16} = 0$ and $a_{16} = 1$. However in this case, suppose $t' < 16$ is the largest t such that $b_{t'}$ is non-zero. First note that if a column, say C^j, is zero, then in the column to its right, say C^{j-1}, C^{j-1}_k (for $k = 0$ to $15 - t'$) can take any value (i.e., are free variables), and the rest of the column C^{j-1} can be all zero. Further, the propagation to columns C^{j-2}, C^{j-3} etc. can be rather weak. Similar pathological examples can be cooked up for other values of a_i, b_is.

A similar situation arises when the code is evaluated in the backward direction. The trick is to keep the above free variables few in number, so that the subspace of such *pathological cases* is of a relatively small dimension. This small dimension is absolutely necessary to keep the exhaustive search over this space tractable. One way to get rid of these pathological free variables is to include a term like W_{i-20}, as we do in our code. This in fact gets rid of all the pathological variables in the forward direction and thereby yields a fast expansion. In the backward direction at least one pathological free variable per column remains, and we must search over such subspaces.

5.4 A Lower Bound on the Minimum Distance

In this section we will prove a lower bound on the code described in the introduction, and given by Equation (5.2). We remark that this is a general technique for reducing the problem to smaller dimensional codes. However, if the reduction is to codes with dimensions too large, then a brute force search may not be feasible. On the other hand, if the reduction is to codes which have really low minimum weight, then we will not obtain a good bound.

We will see in Claim 5.4.3 and Claim 5.4.4 that if the polynomials describing the parity check equations (5.5) have a certain algebraic property, namely that the polynomial corresponding to coefficients a_t is irreducible, and does not divide the polynomial corresponding to coefficients b_t, then some key reduced codes have low dimensions. Although these are not necessary conditions, they make a good choice. Similarly, if the coefficients b_1 and b_{15} are both one, then the number of pathological variables per column is small.

We will prove a lower bound on the minimum weight of the code given by Equation (5.2), but projected on the last 64 words. Clearly, the same bound holds for the full 80 words. The reason we focus on the last 64 words is because the recent attacks on hash functions have shown that the weight of the code in early words (the information words, and a few following words) is mostly immaterial (see "message modification technique" in [WYY05c]), and hence the weight in the latter words decides the complexity of the attack. Later, we will lower bound the code in the last 60 and 48 words. Note that because of the change in the parity equations at index 36, the codewords restricted to the last 48 words cannot be described as easily as Equation (5.2).

Since we will be arguing about the weight of this code in the last 64 words, we instead consider the following code $\mathcal{C}64$: Let $W_0, \cdots, W_{15}$ be the message blocks.

Then

$\underline{\mathcal{C}64}$:

for $i = 16$ to 63

$$W_i = \begin{cases} W_{i-3} \oplus W_{i-8} \oplus W_{i-14} \oplus W_{i-16} \oplus ((W_{i-1} \oplus W_{i-2} \oplus W_{i-15}) <<< 1) \\ \qquad\qquad \text{if } 16 \leq i < 20 \\ W_{i-3} \oplus W_{i-8} \oplus W_{i-14} \oplus W_{i-16} \oplus ((W_{i-1} \oplus W_{i-2} \oplus W_{i-15} \oplus W_{i-20}) <<< 1) \\ \qquad\qquad \text{if } 20 \leq i \leq 63 \end{cases} \tag{5.6}$$

We first prove that this is indeed sufficient. Let $\mathcal{C}$ be the code defined by Equation (5.2).

Lemma 5.4.1 *If the code $\mathcal{C}64$ described above has minimum weight at least 82, then $\mathcal{C}$ has minimum weight at least 82 in its last 64 words.*

Proof: Consider any nonzero codeword in $\mathcal{C}$, say $U = \langle U_0, \cdots, U_{79} \rangle$. Denote $X = \langle U_0, \cdots, U_{15} \rangle$ and $Y = \langle U_{16} \cdots, U_{31} \rangle$ and $Z = \langle U_{32} \cdots, U_{79} \rangle$. Therefore $U = \langle X, Y, Z \rangle$. From Equation (5.2) observe that the code $\mathcal{C}$ is completely determined by specifying any consecutive 16 word block provided the block starts anywhere in 0 to 20, since the rest can then be obtained by solving the recurrence relation. We therefore choose to specify $Y = \langle U_{16}, \cdots, U_{31} \rangle$. That is, we treat Y as the message symbols. Note that a fixed choice of Y also fixes X and Z. Following this observation it is now clear that $\langle Y, Z \rangle$ is a codeword in $\mathcal{C}64$.

Assume that the minimum weight of $\mathcal{C}64$ is d. Then we need to show that any non-zero codeword in $\mathcal{C}$, has weight at least d in its last 64 words. This follows provided X being non-zero implies Y is non-zero. However, Y being zero implies X is zero, as X is a linear function of Y.

Therefore the minimum weight of $\mathcal{C}64$ is exactly the minimum weight of code $\mathcal{C}$ in its last 64 words. ■

Theorem 5.4.2 *The code $\mathcal{C}64$ as defined by Equation (5.6) has minimum distance at least 82.*

Proof: Let $d_{\min}$ stand for the minimum weight of the code $\mathcal{C}64$, and since the code is a linear code it suffices to prove that $d_{\min} \geq 82$. From now onwards, we view the codewords of $\mathcal{C}64$ as a matrix that has 32 columns where each column is 64-bit

long. It is easy to see that the code is invariant under column rotations, i.e., moving the column i to $(i+1)$ yields a (possibly new) codeword. Unless otherwise specified, *the arithmetic in the superscript will be modulo 32.*

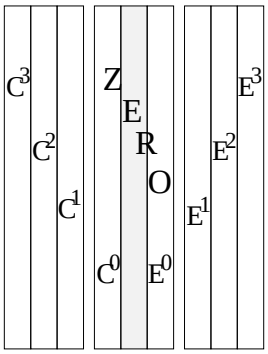
Now consider any non-zero codeword. We break down the proof into two main cases depending upon whether or not a codeword has zero columns.

1. (**All Columns Non-Zero Case:**) Consider any such codeword. Also, consider any non-zero column, w.l.o.g., let it be C^0. Denote the columns, to the left of it by $C^1, C^2, \cdots, C^{31}$. Note that all C^i's are non-zero. In this case, let d_1 denote the minimum weight of this sub-code (that is the set of codewords that do not have a zero column).

   Suppose for any column C^j, there exists an l, such that the combined weight of the columns $C^j, C^{j+1}, \cdots, C^{j+l-1}$ is at least $\mu \cdot l$, then we show that d_1 is at least $(32-(\ell-1)) \cdot \mu + (\ell-1) = 33 \cdot \mu - \ell(\mu-1) - 1$. To see this, we create a partition of the 32 columns into several groups. We pick a non-zero column C^j. Now by assumption there exists ℓ-columns such that the average weight of each column is at least μ. Consider the smallest $\ell' \leq \ell$ that achieves this. Then put these ℓ' columns $C^j, C^{j+1}, \cdots, C^{j+\ell'-1}$ into a group. Call these columns good columns and the group a good group. We then choose $C^{j+\ell'}$ and form another group. We continue like this till no more good groups can be created. The remaining columns are then grouped together. Call this group a bad group. Note that the bad group has average weight at least 1. Now let e be the size of this bad group. Then we have $(32-e)$ good columns. Also by assumption, e could be at most $\ell-1$. Therefore the total weight of the codeword is at least $d_1 \geq \mu \cdot (32-e) + e \geq (32-(\ell-1)) \cdot \mu + (\ell-1) = 33 \cdot \mu - \ell(\mu-1) - 1$. Later, we

2. (**At Least One Column Zero Case:**) Assume that there is at least one zero column. Let d_2 stand for the minimum weight of any such codewords or equivalently, minimum weight of this subcode where the subcode consists of codewords that has at least one zero column. W.l.o.g. we can assume that C^0 is a zero column. Further, w.l.o.g. let C^0 be a zero column such that the column to the left of it is non-zero (note that such a column always exists since we are considering a non-zero codeword and counting is done modulo

32). Denote the columns to the left of C^0 as $C^1, C^2, \cdots$ (see figure).



Also, going towards the right of C^0, denote the first non-zero column by E^1 and thereafter $E^2, E^3, \cdots$. Denote the column to the left of E^1 by E^0. (Note that it may be possible that C^0 and E^0 are the same column.) We argue that a few columns to the left and right of a band of zero columns must contribute a total weight of at least $d_{\min}$.

Also note that w.l.o.g. we can assume that there exists exactly one zero band i.e., a contiguous segment of zero columns. To see this observe that if there is more than one distinct zero band, it is always possible to set the non-zero entries between two zero bands to zero (i.e., consider the parity check matrix, if the entries between a zero band is set to zero, it still satisfies the parity check equation). This can only reduce the minimum weight of the codeword. Therefore we can safely assume that there is exactly one zero band.

Next consider $C^1, C^2, \cdots$. How soon can the sequence yield a zero column, i.e., what is the smallest value of j such that $C^j = E^0$? In order to answer this question, first note that since C^0 is everywhere zero, C^1 is essentially generated by the code whose parity check equations over $\mathbb{F}_2$ are given as follows: Denote $C^1 = \langle y_0, \cdots, y_{63} \rangle$. Then

$$\forall i, 16 \leq i \leq 63, \quad 0 = y_i + y_{i-3} + y_{i-8} + y_{i-14} + y_{i-16}. \tag{5.7}$$

Similarly for a fixed C^1, the column C^2 is generated by the code whose parity check equations over $\mathbb{F}_2$ are given as follows: Denote $C^2 = \langle x_0, \cdots, x_{63} \rangle$. Denote $u_i = y_{i-1} + y_{i-2} + y_{i-15}$. Then

$$0 = \begin{cases} x_i + x_{i-3} + x_{i-8} + x_{i-14} + x_{i-16} + u_i & \text{for } 16 \leq i \leq 19 \\ x_i + x_{i-3} + x_{i-8} + x_{i-14} + x_{i-16} + u_i + y_{i-20} & \text{for } 20 \leq i \leq 63 \end{cases} \tag{5.8}$$

On the other hand E^1 is generated by the code whose parity check equations

over $\mathbb{F}_2$ are given as follows: Denote $E^1 = \langle w_0, \cdots, w_{63} \rangle$. Then

$$0 = \begin{cases} w_{i-1} + w_{i-2} + w_{i-15} & \text{for } 16 \leq i \leq 19 \\ w_{i-1} + w_{i-2} + w_{i-15} + w_{i-20} & \text{for } 20 \leq i \leq 63 \end{cases} \tag{5.9}$$

Similarly for a fixed E^1, the column E^2 is generated by the code whose parity check equations over $\mathbb{F}_2$ are given as follows: Denote $E^2 = \langle z_0, \cdots, z_{63} \rangle$ and $v_i = z_{i-1} + z_{i-2} + z_{i-15}$. Then

$$0 = \begin{cases} w_i + w_{i-3} + w_{i-8} + w_{i-14} + w_{i-16} + v_i & \text{for } 16 \leq i \leq 19 \\ w_i + w_{i-3} + w_{i-8} + w_{i-14} + w_{i-16} + v_i + z_{i-20} & \text{for } 20 \leq i \leq 63 \end{cases} \tag{5.10}$$

We first establish the follow claim.

Claim 5.4.3 *If C^0 is zero, and C^1 is non-zero, then C^2 is non-zero.*

Proof: Assume otherwise, i.e., that C^2 is zero. Consider the 48×64 dimensional parity check matrices H_1 and H_2 (essentially Equation 5.7 and 5.9) over $\mathbb{F}_2$.

$$\begin{pmatrix} 1010000010000100100000 & \cdots & 0000000000000000000 \\ 0101000001000010010000 & \cdots & 0000000000000000000 \\ \ddots & \cdots & \ddots \\ 00000000000000000000 & \cdots & 0101000001000010 01 \end{pmatrix}$$

H_1

$$\begin{pmatrix} 01000000000000011000000 & \cdots & 00000000000000000000 \\ 00100000000000001100000 & \cdots & 00000000000000000000 \\ 00010000000000000110000 & \cdots & 00000000000000000000 \\ 00001000000000000011000 & \cdots & 00000000000000000000 \\ 10000100000000000001100 & \cdots & 00000000000000000000 \\ 01000010000000000000110 & \cdots & 00000000000000000000 \\ \ddots & \cdots & \ddots \\ 00000000000000000000 & \cdots & 10000100000000000000110 \end{pmatrix}$$

H_2

Then we need to show that $H = \begin{pmatrix} H_1 \\ H_2 \end{pmatrix}$ has full rank. To do that it is enough to show that there are 64 linearly independent rows. We consider the 48 rows of H_1 and 16 additional rows, namely the 5^{th} through 20^{th} rows of H_2. We reduce the problem to showing that a certain equation over polynomial ring $\mathbb{F}_2[x]$ does not have solutions in a restricted set of polynomials. We associate with the vector $c = \langle c_0, \cdots, c_{63} \rangle$ in $\mathbb{F}_2^{64}$ the polynomial $c(x) = \sum_{i=0}^{63} c_i x^i$ in $\mathbb{F}_2[x]$. Then the following polynomials can be associated with the 1^{st} and 5^{th} rows of matrices H_1 and H_2, respectively:

$$p(x) \stackrel{\text{def}}{=} x^{16} + x^{13} + x^8 + x^2 + 1,$$

$$r(x) \stackrel{\text{def}}{=} x^{19} + x^{18} + x^5 + 1.$$

Further note that the i^{th} (note $1 \leq i \leq 48$) row of H_1 then gets associated with $x^{i-1}p(x)$. Similarly the j^{th} (note we restrict ourselves to $5 \leq j \leq 20$) row of H_2 then gets associated with $x^{j-5}r(x)$. Therefore, observe that if the 80 rows that we are considering were dependent then we can translate that to a non-zero solution of the following polynomial equation:

$$p(x)\alpha(x) + \beta(x)r(x) = 0,$$

with additional constraints that degree(α) ≤ 47 and degree(β) ≤ 15. However, it is well known that $p(x)$ is irreducible. Therefore, if such a equation holds then it must be the case that $p(x)$ divides $r(x)$. However, it is easy to check that $p(x)$ does not divide $r(x)$, thus leading to a contradiction.
Therefore H has full rank. ■

We now strengthen the claim slightly.

Claim 5.4.4 *If C^0 is zero, and C^1 is non-zero, then C^2, C^3 are non-zero.*

Proof: Consider the following polynomials :

$$p(x) \stackrel{\text{def}}{=} x^{16} + x^{13} + x^8 + x^2 + 1,$$

$$q(x) \stackrel{\text{def}}{=} x^{15} + x^{14} + x,$$

$$r(x) \stackrel{\text{def}}{=} x^{19} + x^{18} + x^5 + 1 = x^4 \cdot q(x) + 1.$$

Let H_1 and H_2 be as above.

First of all note that H_2 has full rank. (This is clear from the matrix. Otherwise, note that we could have an identity

$$q(x) \cdot a(x) + r(x) \cdot b(x) \ = \ 0$$

with degree$(a) \leq 3$ and degree$(b) \leq 43$. Since degree$(q \cdot a) <$ degree(r), this cannot happen.) Now we will show that the rank of the matrix

$$\begin{pmatrix} H_2 & 0 \\ H_1 & H_2 \\ 0 & H_1 \end{pmatrix}$$

is at least 128. Since H_1 has full rank, observe that

$$\begin{pmatrix} H_1 & H_2 \\ 0 & H_1 \end{pmatrix}$$

has rank at least 96. So consider the following 92 independent rows from the above matrix, namely 5^{th} row onwards. We also argue that another additional 5^{th} through 40^{th} rows of the top H_2 are also independent. If not, then they would satisfy the following polynomial equations

$$\begin{aligned} \alpha(x)p(x) + \beta(x)r(x) &= 0 \quad (5.11) \\ x^4\beta(x)p(x) + \gamma(x)r(x) &= 0 \quad (5.12) \end{aligned}$$

with restrictions degree$(\alpha) \leq 47$, degree$(\beta) \leq 43$, and degree$(\gamma) \leq 35$.

Since $p(x)$ is an irreducible polynomial, and $p(x) \nmid r(x)$, observe from Equation (5.11) that $p(x)|\beta(x)$. Hence, set $\beta(x) = \mu(x)p(x)$. Substituting in Equation (5.12) we get

$$x^4 p(x)^2 \mu(x) + \gamma(x)r(x) = 0.$$

Since $p(x)$ is irreducible, and $p(x) \nmid r(x)$, and $x \nmid r(x)$, it must hold that $x^4 p(x)^2 | \gamma(x)$. But that is impossible, since degree$(\gamma) \leq 35 <$ 36 = degree $(x^4 p(x)^2)$. ■

The above proof also highlights that for the rank to be full the recurrence relation must satisfy nice properties. In fact, the following claim strengthens it further.

Claim 5.4.5 *If C^0 is everywhere zero, and C^1 is non-zero, then so is C^2, C^3 and C^4.*

Proof: This claim is proven by checking that the system of equations involved have a full rank, which can be checked by a computer. An algebraic proof of this proof will be of interesting.

In fact the above lemma can further be generalized as follows: Consider the subcode (i.e., set of codewords) where there are at most s ($5 \geq s \geq 4$) consecutive non-zero fixed columns and the rest are filled with zero, then this subcode has dimension exactly $16 \cdot s - 48$. This can be verified by a computer just as for Claim 5.4.5. Then we choose an s_1 so that a search in a space of dimension $16 \cdot s_1 - 48$ is feasible. Thus, we choose $s_1 = 5$. Let d_{21} denote the minimum weight in this subcode, i.e the sub-code which has at most s_1 consecutive non-zero columns (and the rest are all zero).

Now we restrict our attention to the cases where there are at least $s_2 \stackrel{\text{def}}{=} s_1 + 1$ consecutive non-zero columns.

(a) **At least s_2 consecutive columns**: Let $s_f = \lceil \frac{s_2}{2} \rceil$ and consider columns $C^1, C^2, \cdots, C^{s_f}$. Let d_f be the minimum of the combined weight of these columns. (We would like $d_f = 82/2 + \epsilon$.) In particular, we also need to make sure that the dimension of this space (which is at most $16 \times s_f$) is small.

Once we do this, we next consider $s_b = \lfloor \frac{s_2}{2} \rfloor$ columns $E^1, \cdots, E^{s_b}$. Notice that the dimension of this space is small ($16 \times s_b \leq 16 \times s_f$) and hence searchable. However, it turns out that the minimum weight can be extremely small. Fortunately, all these bad cases can be characterized into what we call *pathological cases*. Recall Equation (5.9), the constraints induced on E^1. A quick observation reveals that its free variables are the first 15 bits and the very last bit. If the values taken by E^1s first 15 bits are zero, then we call it a pathological case, and non-pathological otherwise.

i. (**Non-Pathological Case:** i.e., Not all of the first 15 bits of E^1 are zero.) Assuming E^1 to be non-pathological, and let d_b be the minimum of the combined weight of the columns $E^1, \cdots, E^{s_b}$. (We would like $d_b = 82/2 - \epsilon$.) Also, note that by the assumption the columns $C^1, \cdots, C^{s_f}$ and $E^1, \cdots, E^{s_b}$ are all distinct. Thus in the non-pathological case we see any codeword must have weight at least $d_3 = d_f + d_b$.

ii. (**Pathological Case:**) Therefore only the pathological cases remains. This is the most subtle and difficult case. Going back to Equation (5.9), we note that in this case it must hold that $w_{63} = 1$ and for all $0 \leq i \leq 62, w_i = 0$. We call such w *pathological*. Now consider Equation (5.10). We can have two cases here.

In the first case, assume that the first 15 variables of z are zero. In that case, it must hold that $z_{62} = 1$. (Plugging in $i = 16$ to 62 in Equation (5.10) will yield $z_j = 0$ for all $15 \leq j \leq 61$ since $w_i = 0$ for these values.) Also note that z_{63} is free. In this case, we also call z pathological. In fact this may continue along the diagonal i.e., $E^3, E^4, \cdots$ may be pathological. If that happens then it is easy to show that the first non-zero bits of E^3 will be its 61^{st} bit, that of E^4 will be 60^{th} bit and so on. Also each column will have a free variable in its 63^{rd} bit.

In the second case, we assume that not all of its first 15 variables are zero. We call such z's to be non-pathological.

Now in this pathological case we need to consider more columns. Firstly note that it can never be the case that only pathological columns are the non-zero columns. (Otherwise C^1 will be pathological, a contradiction.) In fact it can be argued that there has to be at least 4 non-pathological columns (similar to Claim 5.4.5). Thus this sets an upper bound, say $p_{\max}$, on the number of pathological columns. Thus if we assume there are p pathological columns $E^1, E^2, \cdots, E^p$ and then n non-pathological columns $E^{p+1}, \cdots, E^{p+n}$. Now note that two cases can arise. Either all columns $C^1, \cdots, C^{s_f}$ and $E^1, E^2, \cdots, E^{p+n}$ are distinct or $C^{s_f} = E^{p+n'}$ for some $n' \leq n$.

Case A: In the first case, note that the dimension of the search space

is $16 \times n + p$. The main idea here is to choose n and p appropriately so that the space remains searchable, that is if we increase p by too many (say 16) then we should decrease n (say by 1). If the minimum of the combined weight of the p many pathological and n many non-pathological columns is d_{bp} (we would like it to be $82/2-\epsilon$), then the combined minimum weight in this subcode is at least $d_{pA} = d_f + d_{bp}$. In general, we can consider $p_1, p_2, \cdots, p_l = p_{\max}$ many pathological columns with $n_i = s_b - \gamma_i$ for $i = 1, c \cdots, l$, many non pathological columns, where γ_is are proportionately small (so that search space remains tractable), and if the minimum of the combined weight of these columns is d_{bpAi} (we would like it to be $82/2 - \epsilon$), then the minimum of the combined weight of the corresponding subcode is at least $d_{pAi} = d_f + d_{bpAi}$.

Case B: In the second case, define n be the smallest n' such that $C^{s_f} = E^{p+n'}$. Now consider the subcode where exactly $C^1, C^2, \cdots, C^{s_f} = E^{p+n}, E^{p+n-1}, \cdots, E^1$ columns are non-zero (i.e., fix a set of columns). Then (see Appendix A.3 Claim A.3.1) the nullity of the system can be shown to be

$$p + 64 \times (s_f + n - 1) - 48 \times (s_f + n) = p + 16 \cdot s_f + 16 \cdot n - 64.$$

We employ similar idea as in the previous case. We consider $p_1, p_2, \cdots, p_l = p_{\max}$ many pathological columns along with corresponding $n_i = (s_f + s_b - 1) - \gamma_i$ for $i = 1, \cdots, l$ many non pathological columns (note that C^1, C^2 and C^3 are non pathological and included in this calculation), and let the minimum of the combined weight of these columns be d_{pBi}. (We would like $d_{pBi} \geq 82$ for each $i = 1, \cdots, p_{\max}$.)

Note that this exhausts all possibilities. Thus $d_2 \geq \min\{d_{21}, d_3, d_{pAi}, d_{pBi} | i \in \{1, \cdots, l\}\}$, and $d_{\min} \geq \min\{d_1, d_2\}$.

We choose the parameters carefully and do an exhaustive search. We record the following claim.

Claim 5.4.6 *For any non-zero column C^j, there exists $k, 0 \leq k \leq 7$ such that the combined weight of columns $C^j, C^{j+1}, \cdots, C^{j+k}$ is at least $3 \cdot (k+1)$.*

Proof: This is easily verified by a computer program. We mention that for $k \leq 6$, an average of 3 cannot be assured. ■

Example 5.4.7 *We cite below an example where over 7 columns an average of 3 does not hold. Below we only give 8 columns and the columns are placed horizontally. Note that the 8 columns yield 29, whereas the first 7 columns yield only 14.*

```
00000000000000000000000000000000000000000000000000000001000000
00000000000000000000000000000000000000000000000000000000110110
00000000000000000000000000000000000000000000000000000000010100
00000000000000000000000000000000000000000000000000000000001110
00000000000000000000000000000000000000000000000000000000000100
00000000000000000000000000000000000000000000000000000000000011
00000000000000000000000000000000000000000000000000000000000001
10001010100000000010010000100000100001001011000000100010000010000
```

With this we see that with $\ell = 8$ and $\mu = 3$ we get $d_1 = 82$.

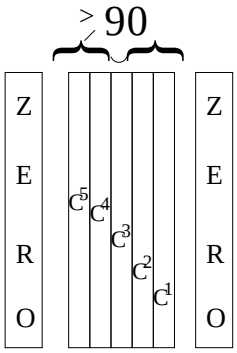
We set $s_1 = 5$ and get $d_{21} = 90$ by an exhaustive search. Note that it is important to choose s_1 small so as to keep the space searchable.

Thus $s_f = 3 = s_b$ and by an exhaustive search, we get $d_f = 42$ (i.e., $\epsilon = 1$) and $d_b = 40$. This implies $d_3 \geq 82$. Recall that this is the non-pathological case, and hence reasonable expansion can be expected.

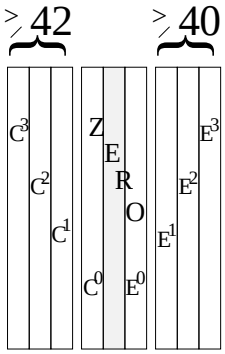
For pathological cases, we first choose $p = 8$ and $n = 3$. We get $d_{bp1} \geq 40$. Similarly, setting $p = 16$ and $n = 2$, we get $d_{bp2} \geq 40$ and setting $p = 28$ and $n = 1$ we get $d_{bp3} \geq 40$. Note in these cases the columns E^{p+n} and C^3 are distinct.

Similarly, for the second subcase of pathological cases, setting $p = 8$, $n = 3$ we get $d_5 = d_{5,1} \geq 82$; setting $p = 16$, $n = 2$ we get $d_{5,2} \geq 82$, and setting $p = 28$, $n = 1$ we get $d_{5,3} \geq 82$. Note that in this case $\exists n'$ s.t. $E^{p+n'} = C^3$. Also note that

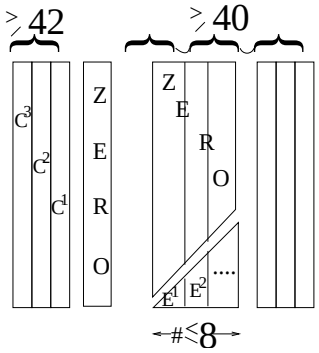
choosing $n = n'$ is sufficient. This completes the proof. ■



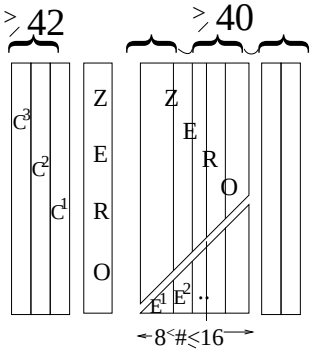
$s_1 = 5, d_2 \geq 90$



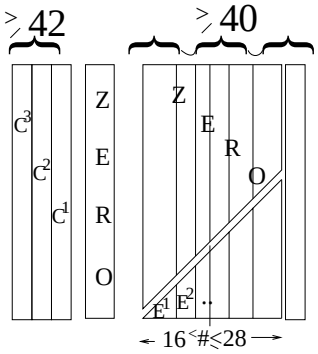
Case 2(a)(i), $s_f = s_b = 3$
non-pathological, $d_f \geq 42, d_b \geq 40$



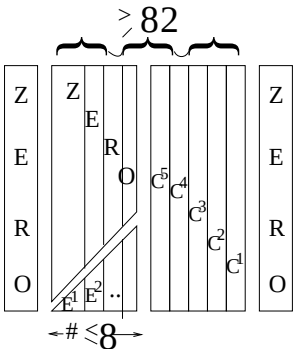
Case 2(a)(ii)(A)(I)



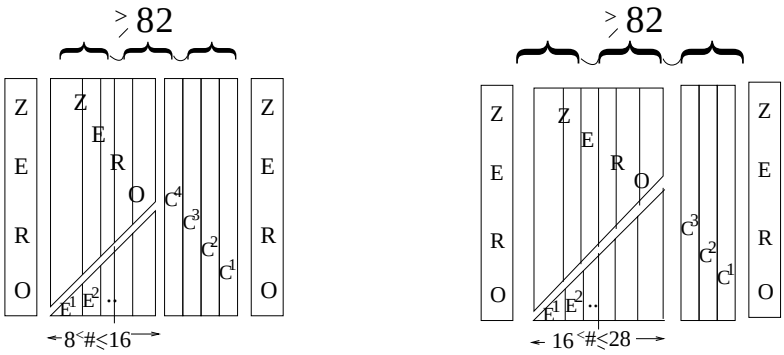
Case 2(a)(ii)(A)(II)



Case 2(a)(ii)(A)(III)



Case 2(a)(ii)(B)(I)



Case 2(a)(ii)(B)(II) Case 2(a)(ii)(B)(III)

Various Cases in the proof of Theorem 5.4.2
(weights referred to the combined weights of the columns)

We remark that the minimum weight of this code can at most be 82 and therefore our result is tight.

Example 5.4.8 *Below is a codeword in the code defined by Equation (5.6) with optimal minimum weight. We found the following codeword while searching for Case 2(b)(ii)(A)(II). Below we only give eight columns that includes six non-zero and two zero columns. The rests are all zero columns. Below the columns are placed horizontally.*

```
0000000000000000 0000000000000000 0000000000000000 0000000000000000
0011110010011110 1000000001101001 1101001001010110 0000110010010000
1011000101000100 0010111101001000 1011100010101100 1101000000101111
1010101000111011 0010100100110010 1000000101001000 0110011000000000
0000000000000000 0000000000000000 0000000000000000 0000000000000100
0000000000000000 0000000000000000 0000000000000000 0000000000000011
0000000000000000 0000000000000000 0000000000000000 0000000000000001
0000000000000000 0000000000000000 0000000000000000 0000000000000000
```

5.5 Further Truncation

In this section, extending our approach we prove that the minimum weight of the code $\mathcal{C}$ in the last 60 words is at least 75 and that in the last 48 words is at least 52, respectively.

Theorem 5.5.1 *The code $\mathcal{C}64$, as defined by Equation (5.6), has minimum weight at least 75 (and at least 52) in its last 60 words (and in its last 48 words, respectively).*

In general, our proof strategy is robust, i.e., it can in principle be adapted to estimate the minimum weight of this code in the last $4 \cdot n$ (where n is an integer) number of steps, though the dimension of the search space increases by an additive factor of $(64 - 4 \cdot n)$ and may make it computationally infeasible. On the other hand, when n gets smaller, say $n \leq 12$, we may only need to show an average 2 per column viz a viz Claim 5.4.6. Since most of our search is conducted using early-stopping, the large dimension is not expected to be a problem.

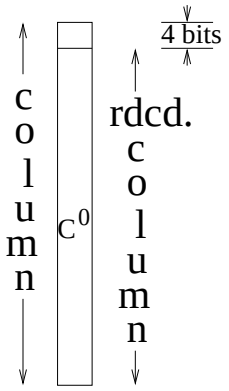
Next, observe that the minimum weight of the code $\mathcal{C}64$ in the last 60 words yields a lower bound on the minimum weight of the code $\mathcal{C}$ in the last 60 words. Reviewing the proof of Theorem 5.4.2, it may be observed that in case 2 (i.e., **At Least One Column Zero Case**) we either consider a codeword (case 2(b)(ii)(A)(II), case 2(b)(ii)(B)(II) and case 2(b)(ii)(C)(II)) or consider few columns (in the remaining cases) which can always be extended to get a valid codeword. Therefore in these cases just counting the weight of the last 60 words gives a lower bound on the minimum weight of the code in the last 60 words. However, the same is not true for case 1 (i.e., **All Columns Non-zero Case**). We handle this case carefully. This then allows us to prove the following theorem.

5.5.1 The Last Sixty Words

Theorem 5.5.2 *The code $\mathcal{C}64$, as defined by Equation (5.6), has minimum weight at least 75 in its last 60 words.*

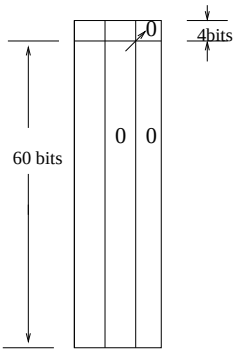
Proof: Consider any column of length 64 bits. A column restricted to its bottom most 60 bits will henceforth be referred to as a *reduced column* (see figure).

Unless otherwise mentioned, we will use the same name, eg., C^0, to denote a column and its reduced column. We divide the proof into three main cases.



A Reduced Column

1. (**All Columns Are Non-zero But Reduced Column Can Be Zero Case**): Consider any such codeword. Also consider any non-zero column, w.l.o.g., let it be C^0. Denote the columns, to the left of C^0 by $C^1, C^2, \cdots, C^{31}$. Note that by assumption all columns are non-zero.
   Then observe that due to this assumption no two consecutive reduced columns can be zero everywhere. To see this let C^0 and C^1 be the columns such that their reduced columns are everywhere zero. Let C^1 be the column left to C^0. Denote C^0 by $x = \langle x_0, x_1, \cdots, x_{63} \rangle$ and C^1 by $y = \langle y_0, y_1, \cdots, y_{63} \rangle$. Note that by the assumption $x_i = y_i = 0$ for all $i = 4, \cdots, 63$. Now consider the parity check equations of $\mathcal{C}64$ and set $i = 20$.



   We get

$$y_{20} + y_{17} + y_{12} + y_6 + y_4 + x_{19} + x_{18} + x_5 + x_0 = 0,$$

   which implies $x_0 = 0$. Similarly by setting $i = 21, 22, 23$, it can be seen that x is everywhere zero.

   We can therefore safely assume that no two consecutive reduced columns are zero. Then, the following can be easily verified by a computer program.

Claim 5.5.3 *For any non-zero column C^i, there exists $k, 0 \leq k \leq 7$ such that the combined weight of the reduced columns $C^i, C^{i+1}, \cdots, C^{i+k}$ is at least $3 \cdot (k+1)$.*

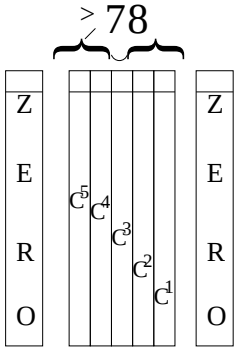
Note that although we restrict ourselves to at most 2 bits ON in reduced C^0, we must consider all 16 possibilities for the first 4 bits of C^0 to be able to define reduced column C^1 (from 16 bits in reduced column in C^1 and all the bits in C^0). Despite this the search is easily conducted.

Then, following the same line of argument as in Case 1 (**All Columns Non-Zero Case**) of Theorem 5.4.2, it can be shown that the total weight of the reduced columns is at least 78. This is because 25 columns yield at least 75 and the remaining seven columns yield at least 3 (since two consecutive reduced columns contribute at least 1).

2. (**At Least One Column Zero Case**): This case can be handled as the **Zero Case** in the proof of theorem 5.4.2. We consider the same number of cases and we count only the last 60 bits in a column. We skip the details and summarize below the results we obtain.

   (a) **Number Of Consecutive Non-Zero Columns Is At Most Five**:

   The combined weight of the 5 non-zero column is then at least 78.

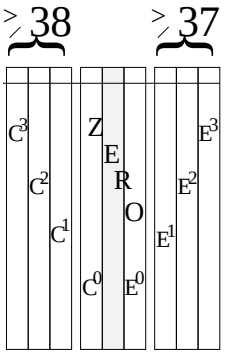


Case 3(a)

   (b) **Number Of Consecutive Non-Zero Columns Is At Least Six**: The combined weight of three reduced columns to the left of a zero band is at least 38.

      i. (**Non-Pathological Case**) The combined weight of three reduced columns to the right of a zero band is at least 38.

Therefore the combined weight of three reduced columns to the left of a zero column and that of three reduced columns to the right of a zero column yields (assuming they are distinct) at least 75.



Case 3(b)(i)

ii. (**Pathological Case**)

A. **# of Pathological columns** ≤ 8

(I). **6^{th} and earlier non-pathological columns are non-zero** : The combined weight of the pathological reduced columns and the first three non-pathological reduced columns to the right of the pathological columns is at least 37.

(II). **6^{th} or earlier non-pathological column is zero**: The combined minimum weight of these reduced columns is at least 75.

B. $8 <$ **# of Pathological columns** ≤ 16

(I). **5^{th} and earlier non-pathological columns are non-zero** : The combined weight of the pathological reduced columns and the first two non-pathological reduced columns to the right of the pathological columns is at least 37.

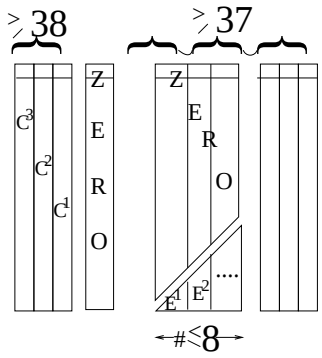
(II). **5^{th} or earlier non-pathological column is zero**: The combined minimum weight of these reduced columns is at least 75.

C. $16 <$ **# of Pathological columns** ≤ 28

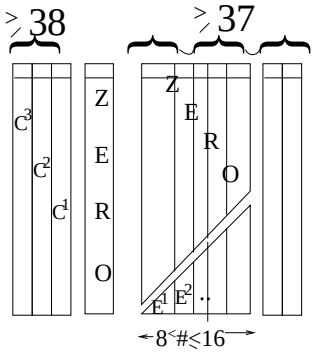
(I). **4^{th} and earlier non-pathological columns are non-zero** : The combined weight of the pathological reduced columns and the first non-pathological reduced columns to the right of the pathological columns is at least 37.

(II). **4^{th} or earlier non-pathological column is zero**: The combined minimum weight of these reduced columns is at least 75.

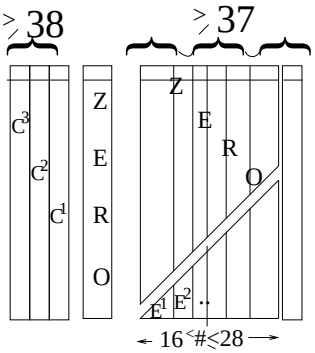
Therefore, in all these cases the combined weight of the reduced column is at least 75. This establishes the theorem. ■



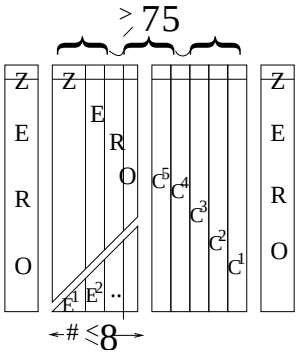
Case 2(b)(ii)(A)(I)



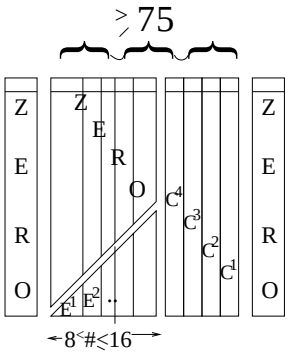
Case 2(b)(ii)(B)(I)



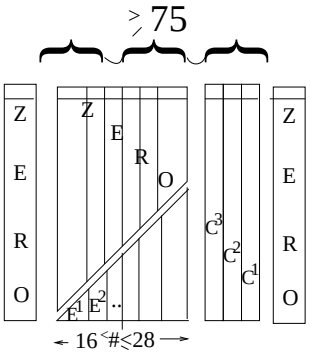
Case 2(b)(ii)(C)(I)



Case 2(b)(ii)(A)(II)



Case 2(b)(ii)(B)(II)



Case 2(b)(ii)(C)(II)

Various Cases in the proof of Theorem 5.5.2
(weights referred to the combined weights of the reduced columns)

Note that our result is tight. The codeword we cite in the previous section achieves this bound.

5.5.2 The Last Forty-eight Words

In this subsection, we prove that the code $\mathcal{C}64$ has minimum weight at least 52 in its last 48 words. As mentioned previously, this proof is more computation intensive as the dimension of the search space increases by an additive factor of 16. The good thing is that we need to show an average 2 per column, viz a viz Claim 5.4.6. This makes our search, conducted using early-stopping, feasible in spite of the apparent large dimension.

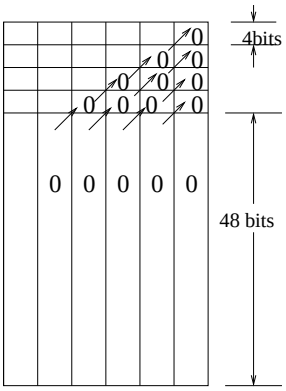
It is easy to observe that the minimum weight of the code $\mathcal{C}64$ in the last 48 words yields a lower bound on the minimum weight of the code $\mathcal{C}$ in the last 48 words. The proof uses the same technique as in the proof of Theorem 5.5.2. Recall that in that proof (that is the proof of Theorem 5.5.2) there are cases where we either consider a codeword or consider few columns which can always be extended to get a valid codeword. In those cases, just counting the weight of the last 48 words suffices to give a lower bound on the minimum weight of the code in the last 48 words. In the remaining case, mimicking the proof of Theorem 5.5.2, we consider reduced columns (here restricted to last 48 entries). We then can verify that under the assumption that all columns are non-zero, the reduced columns cannot be too sparse. This then allows us to prove the following theorem.

Theorem 5.5.4 *The code $\mathcal{C}64$ as defined by Equation (5.6) has minimum weight at least 52 in its last 48 words.*

Proof: Consider any column of length 64 bits. Here a column restricted to its bottom most 48 bits will henceforth be referred as a *reduced column*.
Unless otherwise mentioned, we will use the same name, eg., C^0, to denote a column and its reduced column. We divide the proof into two main cases, depending on the existence of a zero column.

1. (**All Columns Are Non-Zero But Reduced Column Can Be Zero Case**): Consider any such codeword. Also consider any non-zero reduced column, w.l.o.g., let it be C^0. Denote the reduced columns, to the left of C^0 by $C^1, C^2, \cdots, C^{31}$. Note that if five consecutive reduced columns are zero, then the first column must be everywhere zero.

This is easily obtained by setting i suitably in the parity check equations of the code $\mathcal{C}64$ (see figure). We handle that case latter. Therefore we can safely assume that no five consecutive reduced columns are zero.



Then the following is easily verified by a computer program.

Claim 5.5.5 *For any non-zero column C^i, there exists $k, 0 \leq k \leq 6$ such that the combined weight of the reduced columns $C^i, C^{i+1}, \cdots, C^{i+k}$ is at least $(k+1)$. Furthermore, there exists $\ell, 0 \leq \ell \leq 8$ such that the combined weight of the reduced columns $C^i, C^{i+1}, \cdots, C^{i+\ell}$ is at least $2 \cdot (\ell + 1)$.*

Note that although we restrict ourselves to at most 1 bit ON in reduced C^0, we must consider all 2^{16} possibilities for the first 16 bits of C^0 to be able to define reduced column C^1 (from 16 bits in reduced column in C^1 and all the bits in C^0). Since we rely heavily on early stopping, these bits must be guessed in a lazy fashion to make the search feasible. Then following the same line of argument as in Case 1 (**All Columns Non-Zero Case**) of Theorem 5.5.2, it can be shown that the total weight of the reduced columns is at least 53 (since 24 columns yield at least 48 and the remaining eight columns yield at least 8, or 25 columns yield at least 50 and the remaining 7 yields 7, or 26 columns yield 52 and remaining 6 at least 1).

2. **At Least One Column Zero Case**: In this case the first column must be everywhere zero. This case can then be handled as the **Zero Case** in the proof of theorem 5.4.2. We consider the same number of cases and we count only the last 48 bits in a column. We remark that in each such cases, it can be shown that the weight in the last 48 rounds is at least 52. We skip the details.

5.6 Lower Bound for the SHA-1 Code

We now demonstrate that a simple variants of the above technique can be used to give a lower bound on the minimum weight of SHA-1. Specifically we have the following theorem.

Theorem 5.6.1 *SHA-1 message expansion code has minimum weight 25 in the last 60 words.*

Proof: First observe that it suffices to consider the code of length 60 given by the recurrence relation

$$\text{for } i = 16 \text{ to } 59 \quad W_i = (W_{i-3} \oplus W_{i-8} \oplus W_{i-14} \oplus W_{i-16}) <<< 1.$$

We view each codeword as a matrix consisting of 32 columns, each of length 60. Note that the code is invariant under column rotations.

Now if a codeword has all columns non-zero, we are done, as that gives minimum weight at least 32. So, assume that the codeword has one or more zero columns and at least one non-zero column.

Let the column C^1 be the first non-zero column to the right of a band of zero columns. Let the column C^1 be represented by the vector $\langle x_i \rangle_{i=0}^{59}$. Then x satisfies

$$\text{for } i = 16 \text{ to } 59 \quad x_{i-3} \oplus x_{i-8} \oplus x_{i-14} \oplus x_{i-16} = 0,$$

which can be rewritten as :

$$\text{for } i = 13 \text{ to } 56 \quad x_i \oplus x_{i-5} \oplus x_{i-11} \oplus x_{i-13} = 0. \tag{5.13}$$

Thus for any choice of the first 13 bits of x (i.e., $i = 0$ to 12), the bits from $i = 13$ to 56 are determined by the above recurrence. The bits x_{57}, x_{58} and x_{59} are independent, and can be chosen independently.

Similarly, let C^2 be the column to the right of C^1, and let the column be denoted by vector y. Then,

$$\text{for } i = 16 \text{ to } 59 \quad y_{i-3} \oplus y_{i-8} \oplus y_{i-14} \oplus y_{i-16} = x_i,$$

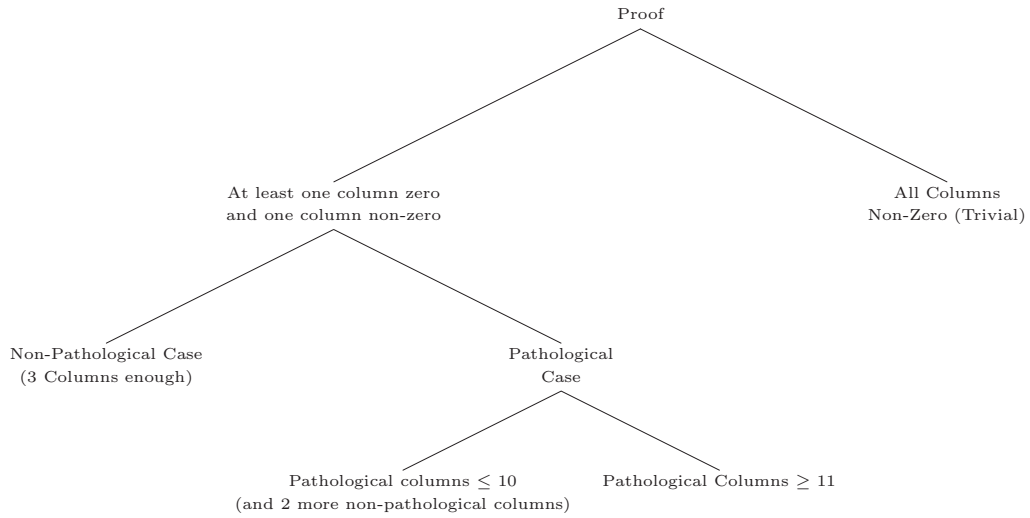
which can be rewritten as

$$\text{for } i = 13 \text{ to } 56 \quad y_i \oplus y_{i-5} \oplus y_{i-11} \oplus y_{i-13} = x_{i+3}. \tag{5.14}$$

Again, given the full vector x, and the first 13 bits of y, the remaining bits of y are given by this relation (except the last three bits, which remain independent). We continue like this to the next column C^3, with z denoting the vector. We mention that if the first 13 bits of x are non zero, then the code expands fast, that is the individual weight of x and y are reasonably good.

So, ideally, we would like to show that no matter how one chooses those bits in x, and in y, and in z, the total weight in the three columns is at least 25. (Of course, we stop early, if just two columns sufficed.) However this is not always true, as C^1 which is required to be the first non-zero column could be *pathological* in the sense that its first 13 bits can be all zero, and hence the bits from $i = 13$ to 56 can also be all zero, and the only non-zero entries come from x_{57}, x_{58} or x_{59}. We call such a column pathological. Similarly, given that C^1 is pathological, C^2 can also be pathological, with non-zero entries in only its last 6 entries this time, and so on.
We now break the proof into two cases based on the values taken by the first 13 bits of C^1 (recall C^1 is the first non-zero column to the right of a band of zero columns).

1. (**Non-pathological Case**): Assume C^1 is non-pathological, that is not all of its first 13 bits are zero. Then by a computer program it can easily be verified that the combined weight of Columns C^1, C^2 and C^3 is at least 25.



2. (**Pathological Case**): Assume C^1 is pathological i.e., each of its first 13 bits is zero. We now make the following easy claim.

   Claim 5.6.2 *If $C^1 = \langle x_i \rangle_{i=0}^{59}$ is pathological, then $x_0 = x_1 = \cdots = x_{56} = 0$.*

   Proof: Since $x_0 = x_1 = \cdots = x_{12} = 0$ (by definition), setting $i = 13$ in Equation (5.13) yields $x_{13} = 0$. Similarly setting $i = 14, \cdots, 56$ gives $x_{14} = x_{15} = \cdots = x_{56} = 0$. ■

   Note that a pathological column does not contribute much to the weight of the codeword. Now denote the columns to the right of C^2 by C^3, C^4 and so on. Next consider C^2. Assume for the moment that it is pathological. Then by the same argument as in Claim 5.6.2 (and Equation (5.14)), it holds that $y_0 = y_1 \cdots = y_{53} = 0$ (set $i = 13, \cdots, 56$ and note that $x_i = 0$ for these values). In general, in a sequence of pathological columns (assume for the moment that this sequence has less than 12 columns) the i^{th} pathological column has the first $60 - 3 \cdot i$ entries zero.

   Assume C^{m+1} is the first non-pathological column (if any). So, if there are exactly m (for the moment assume $m \leq 12$) pathological columns, then the column C^{m+1} (note that C^{m+1} cannot be all zero column by Equation (5.14)) must have a nonzero entry in the first $60-3\cdot(m+1)$ entries. This is equivalent to it having a nonzero entry in the first 13 bits. Since otherwise an argument similar to Claim 5.6.2 can be used to show that all the initial $60-3(m+1)$ bits are zero. The good thing is that a non-pathological column has a reasonably good weight. We now divide the remaining proof into two cases based on the number of consecutive pathological columns.

   (a) (**Number of consecutive pathological columns is at most 10**): In this case, we restrict ourselves to the case where there are 10 or less pathological columns. In this case, the combined weight of the pathological columns and at most two following non-pathological columns can be verified by a computer program to be at least 25.

   (b) (**Number of consecutive pathological columns is at least 11**): If there are a sequence of 11 or more pathological columns, then they already contribute more than 25 as verified by a computer search.

Hence 25 is the lower bound on the last 60 words of the SHA-1 message expansion code. ■

For completeness, we outline below the (combined) search pseudo-code for the Case 1 and Case 2(a).

1. Choose the number m of pathological columns ($0 \leq m \leq 10$). For each pathological column choose the last three bits of that column. The other bits are determined by these bits recalling that in the i^{th} column, the first $60 - 3 \cdot i$ bits are zero.

2. Now choose the first 13 bits of the first non-pathological column (and also choose its last three bits). From these bits all its remaining bits can be determined. If the total count is ≥ 25, then go to the next choice in Step (1); otherwise do Step (3).

3. Choose the first 13 bits of the next column (and its last three bits), from which all its other bits can be determined. If the count is ≥ 25, then go to the next choice in Step (1); otherwise do Step (4).

4. Choose the first 13 bits of the next column (and its last three bits), from which all its other bits can be determined. If the count is < 25, output **FAIL**; otherwise goto the next choice in Step (1).

5.7 SHA1-IME: A modified SHA proposal with a provably good code

In joint work with Charanjit S. Jutla, we propose a new hash function SHA1-**IME** (**IME** stands for "**I**mproved **M**essage **E**xpansion"). We use the same state update transformation as in SHA-1 or SHA-0. However, we replace the SHA-1 message expansion code by an equally simple code that has minimum distance provably at least 82, and that moreover in the last 64 words. The code, denoted by $\mathcal{C}$, can be described as follows: Let $M_0, \cdots, M_{15}$ be the input message blocks.

Define $U_i = (W_{i-1} \oplus W_{i-2} \oplus W_{i-15})$. Then

SHA1-IME :
for $i = 0, 1, \cdots, 15, \quad W_i = M_i$ and
for $i = 16$ to 79

$$W_i = \begin{cases} W_{i-3} \oplus W_{i-8} \oplus W_{i-14} \oplus W_{i-16} \oplus (U_i <<< 13) & \text{if } 16 \le i < 36 \\ W_{i-3} \oplus W_{i-8} \oplus W_{i-14} \oplus W_{i-16} \oplus ((U_i \oplus W_{i-20}) <<< 13) & \text{if } 36 \le i \le 79 \end{cases} \tag{5.15}$$

We now briefly describe the state update function used in SHA-1 (for details see [Uni95]). It comprises 80 steps divided into four rounds. Five 32-bits registers, conveniently denoted as A, B, C, D and E, are used. Their initial state is fixed and we denote it by $\langle A_0, B_0, C_0, D_0, E_0 \rangle$ (and in general, $\langle A_i, B_i, C_i, D_i, E_i \rangle$ after i steps). At step i, W_i is used to alter the state of these registers. Each step uses a fixed constant K_i and a bit-wise boolean function f_i that depends on the specific round. Formally,

for $i = 0$ to 79,
$A_{i+1} = W_i + (A_i <<< 5)$
$+ f_i(B_i, C_i, D_i) + E_i + K_i,$
$B_{i+1} = A_i,$
$C_{i+1} = B_i <<< 30,$
$D_{i+1} = C_i,$
$E_{i+1} = D_i,$

Round	Step(i)	$f_i(X, Y, Z)$
1	0-19	$XY \vee \overline{X}Z$
2	20-39	$X \oplus Y \oplus Z$
3	40-59	$XY \oplus XZ \oplus YZ$
4	60-79	$X \oplus Y \oplus Z$

where '$+$' denotes binary addition modulo 2^{32}.

Observe that the code $\mathcal{C}$ in SHA1-**IME** uses a left rotation by 13 bits. However, it is easy to see that as long as the amount of rotation is relatively prime to 32, the code remains the same up to a permutation of its columns. In particular, its minimum weight does not change if the left rotation by 13 is replaced by a left rotation by 1. We further restrict the code to the last 64 words.

In [JP05c] we informally argue that the SHA1-**IME** is resistant to all collision search attacks. We formally argue that at least the present differential collision

attacks following [CJ98] local-collision based approach is ineffective as the weight of the code is too large. In particular we prove that construction of a differential (or disturbance) vector is not feasible.

In [JP05c] we show that the cipher can be expressed as a 4-CSP (constraint satisfaction problem where each constraint involves at most 4-variables). The predicates we use are majority and xor. It is known that there is an NP-complete problem [GJ79] involving these predicates. At present all the general-purpose (randomized) algorithms to solve 4-CSP takes the 1.4^n or more [Sch99, IT03] (Schoning's algorithm does better in the satisfiable instances which is our case of interest). This therefore rules out possibility of collision attacks on SHA1-**IME** using a general purpose algorithm.

5.8 Conclusion

We have shown how lower bounds on the minimum weight of quasicyclic linear codes of dimension $m \times n$ given by parity equations of the form

$$W_i = \sum_{t=1}^{i} a_{it} W_{i-t} + \left(\left(\sum_{t=1}^{i} b_{it} W_{i-t} \right) <<< 1 \right) \qquad \text{for } i \geq n,$$

can be obtained by reducing the problem to the minimum weight of significantly smaller dimensional codes. Note that this equation is more general than Equation 5.5, and Equation 5.2 is of this form rather than the simpler Equation 5.5. In some cases, we obtain the exact minimum weight, including the example codes we considered. An obvious generalization is to consider three or more column mixing (the equation above has only two column mixing), which could lead to codes with even better minimum distance.

A common paradigm for designing hash functions, including MD5[Riv92], SHA-0, SHA-1 and SHA-2[Uni02] is the following: the 512-bit message is first expanded into N words, and then the N words are used as step keys (sometimes known as round keys) in N steps of a (non-linear) block cipher invoked on an initial vector. The output of the block cipher is the output of the compression function. As pointed out in the Introduction, one of the key ingredients of the recent differential attacks on MD5, SHA-0, and SHA-1 has been their poor message expansion (in terms of

minimum weight) into the N words. Also, it is not known how to lower bound the SHA-2 message expansion. Thus, we consider our technique to be an important advance in the design of collision-resistant hash functions.

Appendix

A.1 Omitted Proofs from Section 3.5

Proof of Lemma 3.5.5: We will use I, J, I', J' to denote $(k+1)$ dimensional vectors over $\mathbb{F}_p$. Now note that

$$\begin{aligned}
g_i(y) &= \text{Plurality}_{y_1,\cdots,y_{k+1}\in\mathbb{F}_p^n}[-\sum_{I\in\mathbb{F}_p^{k+1};I\neq\langle 1,0,\cdots,0\rangle} I_1^i f(I_1(y-y_1)+\sum_{t=2}^{k+1} I_t y_t + y_1)] \\
&= \text{Plurality}_{y-y_1,y_2,\cdots,y_{k+1}\in\mathbb{F}_p^n}[-\sum_{I\in\mathbb{F}_p^{k+1};I\neq\langle 0,\cdots,0\rangle} (I_1+1)^i f(I_1(y-y_1)+\sum_{t=2}^{k+1} I_t y_t + y)] \\
&= \text{Plurality}_{y_1,\cdots,y_{k+1}\in\mathbb{F}_p^n}[-\sum_{I\in\mathbb{F}_p^{k+1};I\neq\langle 0,\cdots,0\rangle} (I_1+1)^i f(\sum_{t=1}^{k+1} I_t y_t + y)] \qquad \text{(A.1)}
\end{aligned}$$

Let $Y = \langle y_1, \cdots, y_{k+1}\rangle$ and $Y' = \langle y'_1, \cdots, y'_{k+1}\rangle$. Now note that

$$\begin{aligned}
1-\eta_i &\leq \Pr_{y_1,\cdots,y_{k+1},b}[T_f^i(y_1,\cdots,y_{k+1},b)=0] = \Pr_{y_1,\cdots,y_{k+1},b}[\sum_{I\in\mathbb{F}_p^{k+1}} I_1^i f(b+I\cdot Y)=0] \\
&= \Pr_{y_1,\cdots,y_{k+1},b}[f(b+y_1)+\sum_{I\in\mathbb{F}_p^{k+1};I\neq\langle 1,0,\cdots,0\rangle} I_1^i f(b+I\cdot Y)=0] \\
&= \Pr_{y_1,\cdots,y_{k+1},y}[f(y)+\sum_{I\in\mathbb{F}_p^{k+1};I\neq\langle 1,0,\cdots,0\rangle} I_1^i f(y-y_1+I\cdot Y)=0] \\
& \qquad \text{(A.2)}
\end{aligned}$$

$$= \Pr_{y_1,\cdots,y_{k+1},y}[f(y) + \sum_{I \in \mathbb{F}_p^{k+1}; I \neq \langle 0,\cdots,0\rangle} (I_1+1)^i f(y + I \cdot Y) = 0]$$

Denote $\langle 0, \cdots, 0\rangle$ by $\vec{0}$. Then for any given $I \neq \vec{0}$ we have the following:

$$\Pr_{Y,Y'}[f(y + I \cdot Y) = \sum_{J \in \mathbb{F}_p^{k+1}; J \neq \vec{0}} -(J_1+1)^i f(y + I \cdot Y + J \cdot Y')] \geq 1 - \eta_i$$

and for any given $J \neq \vec{0}$,

$$\Pr_{Y,Y'}[f(y + J \cdot Y') = \sum_{I \in \mathbb{F}_p^{k+1}; I \neq \vec{0}} -(I_1+1)^i f(y + I \cdot Y + J \cdot Y')] \geq 1 - \eta_i.$$

Combining the above two and using the union bound we get,

$$\begin{aligned} \Pr_{Y,Y'}[\sum_{I \in \mathbb{F}_p^{k+1}; I \neq \vec{0}} (I_1+1)^i f(y + I \cdot Y) &= \\ \sum_{I \in \mathbb{F}_p^{k+1}; I \neq \vec{0}} \sum_{J \in \mathbb{F}_p^{k+1}; J \neq \vec{0}} -(I_1+1)^i (J_1+1)^i f(y + I \cdot Y + J \cdot Y') &= \\ \sum_{J \in \mathbb{F}_p^{k+1}; J \neq \vec{0}} (J_1+1)^i f(y + J \cdot Y')] &\geq \\ 1 - 2(p^{k+1} - 1)\eta \geq 1 - 2p^{k+1}\eta_i & \qquad \text{(A.3)} \end{aligned}$$

The lemma now follows from the observation that the probability that the same object is drawn from a set in two independent trials lower bounds the probability of drawing the most likely object in one trial: Suppose the objects are ordered so that p_i is the probability of drawing object i, and $p_1 \geq p_2 \geq \cdots$. Then the probability of drawing the same object twice is $\sum_i p_i^2 \leq \sum_i p_1 p_i \leq p_1$. ■

Proof of Claim 3.5.16:

$$
\begin{aligned}
T_g(Y,b) &= \sum_{I\in\mathbb{F}_3^{k+1}} g(I\cdot Y+b)\\
&= \sum_{I\in\mathbb{F}_3^{k+1}} [-T_f(I\cdot Y+b-I\cdot Z_1-r_1, I\cdot Z_2+r_2,\cdots, I\cdot Z_{k+1}+r_{k+1},\\
&\quad I\cdot Z_1+r_1)+f(I\cdot Y+b)]\\
&= -\sum_{I\in\mathbb{F}_3^{k+1}}\left[\left[\sum_{\emptyset\neq J'\in\mathbb{F}_3^k} f(I\cdot Y+b+\sum_{t=2}^{k+1}J_tI\cdot Z_t+\sum_{t=2}^{k+1}J_tr_t)\right]\right.\\
&\quad +\left[\sum_{J'\in F_3^k}\left(f(2I\cdot Y+2b-I\cdot Z_1-r_1+\sum_{t=2}^{k+1}J_tI\cdot Z_t+\sum_{t=2}^{k+1}J_tr_t)\right.\right.\\
&\quad \left.\left.\left.+f(I\cdot Z_1+r_1+\sum_{t=2}^{k+1}J_tI\cdot Z_t+\sum_{t=2}^{k+1}J_tr_t)\right)\right]\right]\\
&= -\sum_{0\neq J'\in\mathbb{F}_3^k}\left[\sum_{I\in\mathbb{F}_3^{k+1}} f(I\cdot Y+b+\sum_{t=2}^{k+1}J_tr_t+\sum_{t=2}^{k+1}J_tI\cdot Z_t)\right]\\
&\quad -\sum_{J'\in\mathbb{F}_3^k}\left[\left[\sum_{I\in\mathbb{F}_3^{k+1}} f(2I\cdot Y+2b-I\cdot Z_1-r_1+\sum_{t=2}^{k+1}J_tI\cdot Z_t+\sum_{t=2}^{k+1}J_tr_t)\right]\right.\\
&\quad \left.+\left[\sum_{I\in\mathbb{F}_3^{k+1}} f(I\cdot Z_1+r_1+\sum_{t=2}^{k+1}J_tI\cdot Z_t+\sum_{t=2}^{k+1}J_tr_t)\right]\right]\\
&= \sum_{0\neq J'\in\mathbb{F}_3^k}\left[-T_f(y_1+\sum_{t=2}^{k+1}J_tz_{t,1},\cdots,y_{k+1}+\sum_{t=2}^{k+1}J_tz_{t,(k+1)},b+\sum_{t=2}^{k+1}J_tr_t)\right]\\
&+ \sum_{J'\in\mathbb{F}_3^k}\left[-T_f(2y_1-z_{1,1}+\sum_{t=2}^{k+1}J_tz_{t,1},\cdots,2y_{k+1}-z_{1,(k+1)}+\sum_{t=2}^{k+1}J_tz_{t,(k+1)},\right.\\
&\quad 2b-r_1+\sum_{t=2}^{k+1}J_tr_t)+T_f(z_{1,1}+\sum_{t=2}^{k+1}J_tz_{t,1},\cdots,z_{1,k+1}+\sum_{t=2}^{k+1}J_tz_{t,(k+1)},\\
&\quad \left.r_1+\sum_{t=2}^{k+1}J_tr_t)\right] \qquad \text{(A.4)}
\end{aligned}
$$

■

Proof of Claim 3.5.17:

$$
\begin{aligned}
T^1_{g_1}(Y,b) &= \sum_{I\in\mathbb{F}_3^{k+1}} I_1 g_1(I\cdot Y+b)\\
&= \sum_{I\in\mathbb{F}_3^{k+1}} I_1\big[-T^1_f(I\cdot Y+b-I\cdot Z_1-r_1, I\cdot Z_2+r_2,\cdots, I\cdot Z_{k+1}+r_{k+1},\\
&\qquad I\cdot Z_1+r_1)+f(I\cdot Y+b)\big]\\
&= -\sum_{I\in\mathbb{F}_3^{k+1}} I_1\left[\left[\sum_{\emptyset\neq J'\in\mathbb{F}_3^k} f(I\cdot Y+b+\sum_{t=2}^{k+1} J_t I\cdot Z_t+\sum_{t=2}^{k+1} J_t r_t)\right]\right.\\
&\quad \left.+\left[\sum_{J'\in F_3^k} f(2I\cdot Y+2b-I\cdot Z_1-r_1+\sum_{t=2}^{k+1} J_t I\cdot Z_t+\sum_{t=2}^{k+1} J_t r_t)\right]\right]\\
&= -\sum_{0\neq J'\in\mathbb{F}_3^k}\left[\sum_{I\in\mathbb{F}_3^{k+1}} I_1 f(I\cdot Y+b+\sum_{t=2}^{k+1} J_t r_t+\sum_{t=2}^{k+1} J_t I\cdot Z_t)\right]\\
&\quad -\sum_{J'\in\mathbb{F}_3^k}\left[\sum_{I\in\mathbb{F}_3^{k+1}} I_1 f(2I\cdot Y+2b-I\cdot Z_1-r_1+\sum_{t=2}^{k+1} J_t I\cdot Z_t+\sum_{t=2}^{k+1} J_t r_t)\right]\\
&= \sum_{0\neq J'\in\mathbb{F}_3^k}\left[-T^1_f(y_1+\sum_{t=2}^{k+1} J_t z_{t,1},\cdots, y_{k+1}+\sum_{t=2}^{k+1} J_t z_{t,(k+1)}, b+\sum_{t=2}^{k+1} J_t r_t)\right]\\
&+ \sum_{J'\in\mathbb{F}_3^k}\left[T^1_f(2y_1-z_{1,1}+\sum_{t=2}^{k+1} J_t z_{t,1},\cdots, 2y_{k+1}-z_{1,(k+1)}+\sum_{t=2}^{k+1} J_t z_{t,(k+1)},\right.\\
&\quad \left. 2b-r_1+\sum_{t=2}^{k+1} J_t r_t)\right]
\end{aligned}
\tag{A.5}
$$

■

Proof of Claim 3.5.19:

$$
\begin{aligned}
T^i_{g_i}(Y,b) &= \sum_{I\in\mathbb{F}_p^{k+1}} I^i_1 g_i(I\cdot Y+b)\\
&= \sum_{I\in\mathbb{F}_p^{k+1}} I^i_1\big[-T^i_f(I\cdot Y+b-I\cdot Z_1-r_1, I\cdot Z_2+r_2,\cdots, I\cdot Z_{k+1}+r_{k+1},\\
&\quad I\cdot Z_1+r_1)+f(I\cdot Y+b)\big]\\
&= -\sum_{I\in\mathbb{F}_p^{k+1}} I^i_1\left[\left[\sum_{\emptyset\neq J'\in\mathbb{F}_p^k} f(I\cdot Y+b+\sum_{t=2}^{k+1} J_t I\cdot Z_t+\sum_{t=2}^{k+1} J_t r_t)\right]\right.\\
&+ \left.\left[\sum_{J_1\in\mathbb{F}_p, J_1\neq 1} J^i_1\left[\sum_{J'\in\mathbb{F}_p^k} f(J_1 I\cdot Y+J_1 b-(J_1-1)I\cdot Z_1-(J_1-1)r_1+\right.\right.\right.\\
&\quad \left.\left.\left.\sum_{t=2}^{k+1} J_t I\cdot Z_t+\sum_{t=2}^{k+1} J_t r_t)\right]\right]\right]\\
&= -\sum_{0\neq J'\in\mathbb{F}_p^k}\left[\sum_{I\in\mathbb{F}_p^{k+1}} I^i_1 f(I\cdot Y+b+\sum_{t=2}^{k+1} J_t r_t+\sum_{t=2}^{k+1} J_t I\cdot Z_t)\right]\\
&- \sum_{J'\in\mathbb{F}_p^k}\left[\sum_{J_1\in\mathbb{F}_p; J_1\neq 1} J^i_1\left[\sum_{I\in\mathbb{F}_p^{k+1}} I^i_1 f(J_1 I\cdot Y+J_1 b-(J_1-1)I\cdot Z_1-(J_1-1)r_1\right.\right.\\
&\quad \left.\left.+\sum_{t=2}^{k+1} J_t I\cdot Z_t+\sum_{t=2}^{k+1} J_t r_t)\right]\right]\\
&= \sum_{0\neq J'\in\mathbb{F}_p^k}\left[-T^i_f(y_1+\sum_{t=2}^{k+1} J_t z_{t,1},\cdots, y_{k+1}+\sum_{t=2}^{k+1} J_t z_{t,(k+1)}, b+\sum_{t=2}^{k+1} J_t r_t)\right]\\
&+ \sum_{J'\in\mathbb{F}_p^k}\left[\sum_{J_1\in\mathbb{F}_p; J_1\neq 1} J^i_1\left[-T^i_f(J_1 y_1-(J_1-1)z_{1,1}+\sum_{t=2}^{k+1} J_t z_{t,1},\cdots,\right.\right.\\
&\quad J_1 y_{k+1}-(J_1-1)z_{1,(k+1)}\\
&\quad \left.\left.+\sum_{t=2}^{k+1} J_t z_{t,(k+1)}, J_1 b-(J_1-1)r_1+\sum_{t=2}^{k+1} J_t r_t)\right]\right] \qquad \text{(A.6)}
\end{aligned}
$$

■

A.2 Omitted Proofs from Section 3.7

Lemma A.2.1 *([Sam07]) For a function $f : \{0,1\}^n \to \mathbb{R}$ it holds $\|f\|_{U_3}^8 = \mathbb{E}_y \sum_\alpha \hat{f}_y^4(\alpha)$.*

Lemma A.2.2 *([Sam07])*

$$\mathbb{E}_{xy} \sum_{\alpha\beta} \hat{f}_x^2(\alpha)\hat{f}_y^2(\beta)\hat{f}_{x+y}^2(\alpha+\beta) = \mathbb{E}_y \sum_\alpha \hat{f}_y^6(\alpha).$$

Proof: By simply expanding it can be verified that $(f_x * f_x)(s) = (f_s * f_s)(x)$. Now we consider the expression

$$\mathbb{E}_{xy} \sum_{\alpha\beta} \hat{f}_x^2(\alpha)\hat{f}_y^2(\beta)\hat{f}_{x+y}^2(\alpha+\beta)$$

$$= \mathbb{E}_{xy} \sum_{\alpha\beta} \mathbb{E}_{uu'vv'zz'} f_x(u)f_x(u')\chi_\alpha(u+u')f_y(v)f_y(v')\chi_\beta(v+v')f_{x+y}(z)f_{x+y}(z')\chi_{(\alpha+\beta)}(z+z')$$

$$= \mathbb{E}_{xystruvz} \sum_{\alpha\beta} f_x(u)f_x(u+s)\chi_\alpha(s)f_y(v)f_y(v+t)\chi_\beta(t)f_{x+y}(z)f_{x+y}(z+r)\chi_{(\alpha+\beta)}(r)$$

$$= \mathbb{E}_{xystruvz} f_x(u)f_x(u+s)f_y(v)f_y(v+t)f_{x+y}(z)f_{x+y}(z+r)\delta_r^s\delta_r^t$$

$$= \mathbb{E}_{xysuvz} f_x(u)f_x(u+s)f_y(v)f_y(v+s)f_{x+y}(z)f_{x+y}(z+s) = \mathbb{E}_{xys}(f_x*f_x)(s)(f_y*f_y)(s)(f_{x+y}*f_{x+y})(s)$$

$$= \mathbb{E}_{xys}(f_s * f_s)(x)(f_s * f_s)(y)(f_s * f_s)(x+y) = \mathbb{E}_s \sum_\alpha \widehat{f_s * f_s}^3(\alpha) = \mathbb{E}_x \sum_\alpha \hat{f}_x^6(\alpha) \blacksquare$$

With the above the following corollary is immediate.

Corollary A.2.3

$$\mathbb{E}_{uyz} \sum_{\alpha\beta} \hat{f}_{uy}^2(\alpha)\hat{f}_{uz}^2(\beta)\hat{f}_{u(y+z)}^2(\alpha+\beta) = \mathbb{E}_{uy} \sum_\alpha \hat{f}_{uy}^6(\alpha).$$

Claim A.2.4 *([Sam07]) Define a function $F : \{0,1\}^n \to \mathbb{R}$ by $F(x) = \hat{g}_x^2(Dx)$, where g is an arbitrary boolean function and $D : \{0,1\}^n \to \{0,1\}^n$ is an arbitrary linear map. Then $\widehat{F}(v) = \mathbb{E}_s\, \hat{g}_s^2(D^t s + v)$*

Proof of the Claim:

$$\widehat{F}(v) = \mathbb{E}_z\, F(z)\chi_v(z) = \mathbb{E}_z\, \hat{g}_z^2(Dz)\chi_v(z) = \mathbb{E}_z\mathbb{E}_{yw}\, g_z(y)g_z(w)\chi_{Dz}(y+w)\chi_v(z)$$

$$= \mathbb{E}_{zys}\ g_z(y)g_z(y+s)\chi_{Dz}(s)\chi_v(z) = \mathbb{E}_{zs}\ g_z{*}g_z(s)\chi_{Dz}(s)\chi_v(z) = \mathbb{E}_{zs}\ g_s{*}g_s(z)\chi_{Dz}(s)\chi_v(z)$$

$$= \mathbb{E}_{zs}\ \sum_y \hat{g}_s^2(y)\chi_y(z)\chi_{D^t s}(z)\chi_v(z) = \mathbb{E}_s \sum_y\ \delta^y_{D^t s+v}\ \hat{g}_s^2(y) = \mathbb{E}_s\ \hat{g}_s^2(D^t s+v). \qquad \blacksquare$$

Claim A.2.5 *If $\Gamma_i = \Gamma_i^t$ and $y \cdot \Gamma u$ is invariant under transformation $y \mapsto u, u \mapsto y$, then Γ is invariant under the action of Sym_3.*

Proof: Define $\Lambda_{ijk} \stackrel{\text{def}}{=} \Gamma_{kji}$. Then

$$(y \cdot \Gamma u)_j = \sum_{ijk} y_i \Gamma_{ijk} u_k = \sum_{ijk} y_i \Lambda_{kji} u_k = (u \cdot \Lambda y)_j = (y \cdot \Lambda u)_j.$$

Since this holds for any arbitrary u, y, it must be the case that $\Lambda = \Gamma$ i.e., $\Gamma_{kji} = \Lambda_{ijk} = \Gamma_{ijk}$. Since $\Gamma_{ijk} = \Gamma_{ikj}$, we get

$$\Gamma_{ijk} = \Gamma_{ikj} = \Gamma_{kji} = \Gamma_{kij} = \Gamma_{jik} = \Gamma_{jki},$$

ie., Γ is invariant under the action of Sym_3. (Equivalently, for any bijection $\sigma : \{u, x, y\} \to \{u, x, y\}$, $\langle x, y\Gamma u\rangle = \langle \sigma(x), \sigma(y)\Gamma\sigma(u)\rangle$.) $\blacksquare$

Lemma A.2.6 *Let*

$$\sum_{\alpha\beta} \hat{f}^2(\alpha)\hat{g}^2(\beta)\hat{h}^2(\alpha+\beta) \geq \rho,$$

then

$$\sum_{\alpha\beta} \hat{f}^4(\alpha)\hat{g}^4(\beta)\hat{h}^2(\alpha+\beta) \geq \Omega(\rho^5).$$

Proof: Denote

$$\mathbb{F}_2^n \times \mathbb{F}_2^n \supset S \stackrel{\text{def}}{=} \{(\alpha,\beta) \mid \min\{|\hat{f}(\alpha)|, |\hat{g}(\beta)|, |\hat{h}(\alpha+\beta)|\} < \epsilon\}.$$

Further define

$$S \supset S_1 \stackrel{\text{def}}{=} \{(\alpha,\beta) \mid |\hat{f}(\alpha)| = \min\{|\hat{f}(\alpha)|, |\hat{g}(\beta)|, |\hat{h}(\alpha+\beta)|\} < \epsilon\},$$

$$S \supset S_2 \stackrel{\text{def}}{=} \{(\alpha,\beta) \mid |\hat{g}(\beta)| = \min\{|\hat{f}(\alpha)|, |\hat{g}(\beta)|, |\hat{h}(\alpha+\beta)|\} < \epsilon\},$$

$$S \supset S_3 \stackrel{\text{def}}{=} \{(\alpha,\beta) \mid |\hat{h}(\alpha+\beta)| = \min\{|\hat{f}(\alpha)|, |\hat{g}(\beta)|, |\hat{h}(\alpha+\beta)|\} < \epsilon\}.$$

Denote $\overline{S} = \mathbb{F}_2^n \times \mathbb{F}_2^n - S$. Then

$$\rho \leq \sum_{\alpha\beta} \hat{f}^2(\alpha)\hat{g}^2(\beta)\hat{h}^2(\alpha+\beta) \leq \sum_{i=1}^{3} \sum_{(\alpha,\beta)\in S_i} \hat{f}^2(\alpha)\hat{g}^2(\beta)\hat{h}^2(\alpha+\beta) +$$

$$\sum_{(\alpha,\beta)\in\overline{S}} \hat{f}^2(\alpha)\hat{g}^2(\beta)\hat{h}^2(\alpha+\beta) < 3\epsilon^2 + \sum_{(\alpha,\beta)\in\overline{S}} \hat{f}^2(\alpha)\hat{g}^2(\beta)\hat{h}^2(\alpha+\beta).$$

We set $\epsilon \stackrel{\text{def}}{=} \sqrt{\rho/3} - o(\rho)$. This implies $\overline{S} \neq \emptyset$. Therefore,

$$\sum_{\alpha\beta} \hat{f}^4(\alpha)\hat{g}^4(\beta)\hat{h}^2(\alpha+\beta) \geq \sum_{(\alpha,\beta)\in\overline{S}} \hat{f}^4(\alpha)\hat{g}^4(\beta)\hat{h}^2(\alpha+\beta) \geq \epsilon^{10} = \Omega(\rho^5).$$

■

Claim A.2.7 *Suppose it holds that for all x, y, z and all bijection $\sigma : \{x, y, z\} \to \{x, y, z\}$, it is true that $\langle x, y \cdot \Gamma z\rangle = \langle \sigma x, \sigma y \cdot \Gamma \sigma z\rangle$, then Γ is invariant under the action of Sym_3.*

Proof: Note that since for all x, y, z it holds that $\sum_{ijk} x_i y_j z_k (\Gamma_{ijk} + \Gamma_{ikj}) = 0$, it must hold $\Gamma_{ijk} = \Gamma_{ikj}$. Similarly the other cases. ■

Lemma A.2.8 *Let Q be a d dimensional parallelogram. Further let $\{f^p\}_{p\in Q}$ be a family of boolean functions. Moreover let g be another function. Let*

$$\sum_{\{\alpha_p\}_{p\in Q}} \prod_{p\in Q} \widehat{f^p}^2(\alpha_p) \cdot \hat{g}^2(\sum_p \alpha_p) \geq \rho,$$

then

$$\sum_{\{\alpha_p\}_{p\in Q}} \prod_{p\in Q} \widehat{f^p}^4(\alpha_p) \cdot \hat{g}^2(\sum_p \alpha_p) \geq \left(\frac{\rho}{2^d+1}\right)^{2^{d+1}+1}.$$

Proof: Analogous to Lemma A.2.6. ■

Lemma A.2.9 *If f and g are boolean functions i.e., $f, g : \{0,1\}^n \to \{-1,1\}$, then for all $d \geq 1$,*

$$\mathbb{E}_{x_1\cdots x_d} [\, \mathbb{E}_z\, f_{x_1\cdots x_d}(z) g_{x_1\cdots x_d}(z)\,]^2 = \sum_\alpha \widehat{fg}_\alpha^4.$$

Proof: Denote $h(z) \stackrel{\text{def}}{=} fg(z) \stackrel{\text{def}}{=} f(z)g(z)$ and $F(z) \stackrel{\text{def}}{=} f_{x_1}(z), G(z) \stackrel{\text{def}}{=} g_{x_1}(z)$ and $H(z) \stackrel{\text{def}}{=} FG(z) = F(z)G(z)$. Note that h is a boolean function too. By [Sam07], the above holds for $d = 1$ (and also above, we show that it holds for $d = 2$). Assume it holds for d. Now denote $e \stackrel{\text{def}}{=} d + 1$.

$$\mathbb{E}_{x_1 \cdots x_e} [\, \mathbb{E}_z \, f_{x_1 \cdots x_e}(z) g_{x_1 \cdots x_e}(z) \,]^2 = \mathbb{E}_{x_1} \mathbb{E}_{x_2 \cdots x_e} [\, \mathbb{E}_z F_{x_2 \cdots x_e}(z) G_{x_2 \cdots x_e}(z)]^2 = \mathbb{E}_{x_1} \sum_\alpha \widehat{H}_\alpha^4$$

Now note that

$$\widehat{H}_\alpha = \sum_y H(y)\chi_\alpha(y) = \sum_y \chi_\alpha(y) h(y) h(x_1 + y) = \sum_{y\beta\gamma} \hat{h}_\beta \hat{h}_\gamma \chi_\beta(y) \chi_\gamma(x_1 + y) \chi_\alpha(y)$$

$$= \sum_{\beta\gamma} \hat{h}_\beta \hat{h}_\gamma \delta_\beta^{\alpha+\gamma} \chi_\gamma(x_1) = \sum_\beta \hat{h}_\beta \hat{h}_{\alpha+\beta} \chi_\beta(x_1).$$

Using the above, we deduce

$$\mathbb{E}_{x_1 \cdots x_e} [\, \mathbb{E}_z f_{x_1 \cdots x_e}(z) g_{x_1 \cdots x_e}(z) \,]^2 = \mathbb{E}_{x_1} \sum_\alpha \widehat{H}_\alpha^4 = \mathbb{E}_x \sum_\alpha \left(\sum_\beta \hat{h}_\beta \hat{h}_{\alpha+\beta} \chi_\beta(x) \right)^4$$

$$= \mathbb{E}_x \sum_{\alpha'} \left(\sum_\beta \hat{h}_\beta \hat{h}_{\alpha'} \chi_\beta(x) \right)^4 = \mathbb{E}_x \sum_{\alpha'} \left(\hat{h}_{\alpha'} \sum_\beta \hat{h}_\beta \chi_\beta(x) \right)^4 = \mathbb{E}_x \sum_\alpha \left(\hat{h}_\alpha h(x) \right)^4$$

$$= \sum_\alpha \widehat{fg}_\alpha^4.$$

■

Lemma A.2.10 *$\hat{f}_{x_1 \cdots x_d}(z) = 0$ for any $x_1, \cdots, x_d$ with $\langle \sum_{i \in [d]} x_i, z \rangle = 1$.*

Proof:

$$\hat{f}_{x_1 \cdots x_d}(z) = \mathbb{E}_x f_{x_1 \cdots x_d}(x) \chi_z(x) = \mathbb{E}_x \prod_{J \subseteq [d]} f(x + \sum_{j \in J} x_j) \chi_z(x)$$

$$= \mathbb{E}_y \prod_{J \subseteq [d]} f(y + \sum_{j \in J} x_j) \chi_z(y + \sum_{i \in [d]} x_i) = \chi_z(\sum_{i \in [d]} x_i) \;\; \mathbb{E}_y \prod_{J \subseteq [d]} f(y + \sum_{j \in J} x_j) \chi_z(y)$$

$$= -\hat{f}_{x_1 \cdots x_d}(z)$$

■

A.2.1 A Robust Characterization of Bilinearity

A function $f : \mathbb{F}_2^n \times \mathbb{F}_2^n \rightarrow \mathbb{F}_2^m$ is said to be bilinear if it satisfies

$$\forall u, x, y, \in \mathbb{F}_2^n \quad (i) f(x,u)+f(y,u) = f(x+y,u) \quad \text{and} \quad (ii) f(u,x)+f(u,y) = f(u,x+y).$$

It can be shown that then $f(x,y) = \sum_{ij} a_{ij} x_i y_j$, where $a_{ij} \in \mathbb{F}_2^m$.

Define $f_{,z}(u,v) \stackrel{\text{def}}{=} f(u,v+z)+f(u,v)$ and similarly, $f_{z,}(u,v) \stackrel{\text{def}}{=} f(u+z,v)+f(u,v)$.

Lemma A.2.11 *(Exact Characterization A) A function is bilinear iff $f_{x,y}(u,v) = f(x,y)$.*

Proof: Setting $x = y = 0$, we obtain $f(0,0) = 0$. Setting $x = u$ and $y = v$, we obtain $f(0,v) = f(u,0)$. Setting $x = u$ and $y = 0$, we obtain $f(u,0) = 0$. Hence $f(v,0) = 0$. Now setting $u = 0$ yields $f(x,y) + f(x,v) = f(x,v+y)$. Similarly, setting $v = 0$ yields $f(x,y) + f(u,y) = f(x+u,y)$. Thus f is bilinear. ■

We first do a Rubinfeld-Sudan [RS96] type of analysis. Define

$$g(x,y) \stackrel{\text{def}}{=} \text{Plurality}_{u,v \in \mathbb{F}_2^n} \{ f_{x,y}(u,v) \}$$

$$= \text{Plurality}_{u,v \in \mathbb{F}_2^n} \{ f(u+x,v+y) + f(u+x,v) + f(u,v+y) + f(u,v) \}$$

Further define $\eta = \Pr_{u,v,x,y} [\, f_{x,y}(u,v) + f(x,y) \neq 0 \,]$. Define the distance between two functions to be the probability that they disagree i.e.,

$$\text{dist}(f,g) \stackrel{\text{def}}{=} \Pr_{(x,y) \in (\mathbb{F}_2^n)^2} [\, f(x,y) \neq g(x,y) \,]$$

Lemma A.2.12 *Then dist$(f,g) \leq 2\eta$.*

Proof: By a simple averaging argument. ■

Lemma A.2.13 *For all $x, y, \Pr_{u,v} [\, g(x,y) = f_{x,y}(u,v) \,] \geq 1 - 8\eta$.*

Proof: Note that

$$\Pr_{u,u',v,v'} \left[\, f(u+x,v+y) = f_{u+x,v+y}(u',v') \,\right] \geq 1 - \eta,$$

$$\Pr_{u,u',v,v'}\left[\, f(u+x,v) = f_{u+x,v}(u',v') \,\right] \geq 1-\eta,$$

$$\Pr_{u,u',v,v'}\left[\, f(u,v+y) = f_{u,v+y}(u',v') \,\right] \geq 1-\eta,$$

$$\Pr_{u,u',v,v'}\left[\, f(u,v) = f_{u,v}(u',v') \,\right] \geq 1-\eta,$$

and similar things for $f(u',v')$ etc. Therefore,

$$\Pr_{u,u',v,v'}\left[\, f_{x,y}(u,v) = f_{x,y}(u',v') \,\right] \geq 1-8\eta.$$

Since the collision probability of a distribution gives a lower bound to the maximum probability of the distribution, this proves the lemma. ■

Lemma A.2.14 *If $\eta < \frac{1}{40}$, then g is a bilinear function.*

Proof: Fix x, y, u, v. Now observe that each of the following events hold with probability (when u', v' are chosen uniformly at random) at least $1 - 8\eta$:

$$E_1 : g(x,y) = f_{x,y}(u'+u, v'+v); \quad E_2 : g(u,v) = f_{u,v}(u',v');$$

$$E_3 : g(u+x,v) = f_{u+x,v}(u',v'); \quad E_4 : g(u,v+y) = f_{u,v+y}(u',v');$$

$$E_5 : g(u+x,v+y) = f_{u+x,v+y}(u',v').$$

Thus with probability at least $1 - 40\eta$, all of them happen, and in that case it is easy to verify that

$$g(x,y) = g(u,v) + g(u+x,v) + g(u,v+y) + g(u,v).$$

In particular, if $\eta < 1/40$, then $g(x,y) = g_{x,y}(u,v)$ for all x, y, u, v, i.e., g is bilinear. ■

Following BLR (and BCLR) analysis, the above lemma can be improved slightly (see, [BLR93, BOCLR04]).

Lemma A.2.15 *For $(x,y) \in (\mathbb{F}_2^n)^2$, define $\epsilon_{xy} = \Pr_{u,v}\left[\, g(x,y) \neq f_{x,y}(u,v) \,\right]$ and define $\epsilon = \max_{xy} \epsilon_{xy}$. Then if $\eta < \frac{1}{25}$, then $\epsilon \leq \alpha$ where α is the smallest root of $X^2 - X + 4\eta = 0$.*

Proof: As in [BOCLR04, BLR93], for any given $(x,y) \in (\mathbb{F}_2^n)^2$ it can be shown that

(provided η is small enough as given)

$$1 - 8\eta \leq (1 - \epsilon_{xy})^2 + \epsilon_{xy}^2 \quad \text{i.e.,} \quad 4\eta \geq \epsilon_{xy} - \epsilon_{xy}^2.$$

Furthermore, as $\epsilon_{xy} < 1/2$ for all (x, y), we get that $\epsilon_{xy} \leq \alpha$, where α is the smallest root of $X^2 - X + 4\eta = 0$. In particular, this implies that $\epsilon \leq \alpha$. ■

Lemma A.2.16 *If $\eta < \frac{1}{25}$, and hence, $\epsilon < \frac{1}{5}$, then g is bilinear.*

Proof: From Lemma A.2.15, it is clear that if $\eta < \frac{1}{25}$, then $\epsilon < \frac{1}{5}$. Then as in Lemma A.2.14, for any fixed x, y, u, v, with probability (over the choices of u', v') at least $1 - 5\epsilon$ (and hence, with positive probability) all the events $E_1, \cdots, E_5$ happen. This then implies that g is bilinear. ■

Lemma A.2.17 *It holds dist$(f, g) \leq \eta + \epsilon$. Therefore, dist$(f, g) \leq \min\{2\eta, \eta + \epsilon\}$.*

We modify the exact characterization slightly.

Lemma A.2.18 *(Exact Characterization B) A function is bilinear iff $\forall w, h, x, y, z$ it holds that $f(w, h) = f(x, y) + f(x, y + h) + f(x + w, z) + f(x + w, z + h)$.*

Proof: Easy. ■

With this we redefine

$$g(w, h) \stackrel{\text{def}}{=} \text{Plurality}_{x,y,z \in \mathbb{F}_2^n}\{f(x, y) + f(x, y + h) + f(x + w, z) + f(x + w, z + h)\}.$$

Further redefine

$$\eta = \Pr_{h,w,x,y,z}[\, f(w, h) + f(x, y) + f(x, y + h) + f(x + w, z) + f(x + w, z + h) \neq 0\,].$$

With the above exact characterization, we can prove

Lemma A.2.19 *For all w, h,*

$$\Pr_{x,y,z}[\, g(w, h) = f(x, y) + f(x, y + h) + f(x + w, z) + f(x + w, z + h)\,] \geq 1 - 4\eta.$$

Proof: Observe that when x, x', y, y', z, z' are chosen uniformly randomly, then each of the following events happens with probability at least $1 - \eta$:

$$E_1 : \; f(x + x', y + y' + z + z') = f(x, y) + f(x, y' + z + z') + f(x', y') + f(x', z + z' + y),$$

$$E_2: \quad f(x+x', y+y'+z'+z+h) = f(x, y+h) + f(x, y'+z'+z) + f(x', y'+h)$$
$$++ f(x', y+z'+z),$$
$$E_3: \quad f(x+x', y+y'+z+z') = f(x+w, z) + f(x+w, z'+y+y') + f(x'+w, z')$$
$$+f(x'+w, y+y'+z),$$
$$E_4: \quad f(x+x', y+y'+z+z'+h) = f(x+w, z+h) + f(x+w, z'+y+y')$$
$$+f(x'+w, z'+h) + f(x'+w, y+y'+z).$$

By union bound, the probability that all of them happen is at least $1-4\eta$. However, when all of them happen, then we get

$$f(x,y) + f(x, y+h) + f(x+w, z) + f(x+w, z+h) =$$

$$f(x', y') + f(x', y'+h) + f(x'+w, z') + f(x'+w, z'+h),$$

i.e., a collision. Since collision probability lower bounds the quantity we want, this completes the proof. ■

Remark A.2.20 *The lemma above can be extended easily over abelian groups as follows.*

$$E_1: \quad f(-x+x', -(y+y'+z+z')) = f(x,y) -$$
$$f(x, -(y'+z+z')) - f(x', y') + f(x', -(z+z'+y)),$$
$$E_2: \quad f(-x+x', -(y+y'+z'+z+h)) = f(x, y+h) - f(x, -(y'+z'+z))$$
$$-f(x', y'+h) + f(x', -(y+z'+z)),$$
$$E_3: \quad f(-x+x', -(y+y'+z+z')) = f(x+w, z) - f(x+w, -(z'+y+y'))$$
$$-f(x'+w, z') + f(x'+w, -(y+y'+z)),$$
$$E_4: \quad f(x+x', -(y+y'+z+z'+h)) = f(x+w, z+h) - f(x+w, -(z'+y+y'))$$
$$-f(x'+w, -(z'+h)) + f(x'+w, -(y+y'+z)).$$

Further note that if the maps $2: G_1 \to G_1$ and $2: G_2 \to G_2$, where $2(x) = x + x$, induce isomorphisms, then one can replace the above by BLR's extension (i.e., one can take $y = z$).

The above lemma can easily be seen to imply the next lemma.

Lemma A.2.21 *If $\eta < \frac{2}{25}$, and hence, $\epsilon < \frac{1}{5}$, then g is bilinear.*

Proof: Firstly observe that we can slightly improve Lemma A.2.15 as follows (with the same definition of ϵ_{xy} and ϵ as in there): If $\eta < \frac{2}{25}$, then $\epsilon \leq \alpha$ where α is the smallest root of $X^2 - X + 2\eta = 0$. This follows as we can now show

$$1 - 4\eta \leq (1 - \epsilon_{xy})^2 + \epsilon_{xy}^2.$$

The rest is similar to Lemma A.2.16. ■

A.3 A Rank Proof

Recall that we used E^0 to denote a column that is zero everywhere. Also, recall that the columns left to E^0 are denoted E^1, E^2 and so on. In the following claim, we will assume $3 \leq n$.

Claim A.3.1 *Let $E^1, E^2, \cdots, E^p$ be p pathological columns. Also, let $E^{p+1}, E^{p+2}, \cdots, E^{p+n}$ be n non-pathological columns. Further assume that $E^{p+n+1} = C^0$ is everywhere zero. If the nullity of the parity check equations resulting from these columns with $p = 0$ is $16 \cdot n - 48$, then the nullity of the parity check equations resulting from these columns with any $p \leq 28$ is*

$$p + 16 \cdot n - 48.$$

Proof: Let $N_{i,j}, (1 \leq i \leq n, 0 \leq j \leq 63)$ denote the entries in the non-pathological columns. Also let $P_{i,j}, (1 \leq i \leq p, \text{ for each } i, \ 64 - i \geq j \leq 63)$ be the pathological variables. We will denote $N_i = \langle N_{i,0}, \cdots, N_{i,63} \rangle$ and $P_i = \langle P_{i,64-i}, \cdots, P_{i,63} \rangle$. Let $H_{1|i}$ denote the matrix H_1 restricted to the last i columns. (Note that only the last i rows will be non-zero.) Also let $H_{2|i}$ denote the matrix H_2 restricted to the last i columns. (Note that only the last $i - 1$ rows will be non-zero.) Note that

$\langle P_1, \cdots, P_p, N_1, \cdots, N_n \rangle$ must belong to the null space of the following matrix:

$$\mathcal{H} = \begin{pmatrix} H_{1|1} & H_{2|2} & & & & & & & & & \\ & H_{1|2} & H_{2|3} & & & & & & & & \\ & & & \ddots & \ddots & & & & & & \\ & & & & & H_{1|p-1} & H_{2|p} & & & & \\ & & & & & & H_{1|p} & H_2 & & & \\ & & & & & & & H_1 & H_2 & & \\ & & & & & & & \ddots & \ddots & & \\ & & & & & & & & & H_1 & H_2 \\ & & & & & & & & & & H_1 \end{pmatrix}$$

Note that when we restrict H_1 or H_2 to the last few columns, the top rows in that restricted entries may become zero row. We remove such rows if the entire row in the above matrix $\mathcal{H}$ becomes everywhere zero. Note that with this modification, the following sub-matrix is already in the echelon form:

$$\mathcal{H}1 = \left.\begin{pmatrix} H_{1|1} & H_{2|2} & & & & \\ & H_{1|2} & H_{2|3} & & & \\ & & & \ddots & \ddots & \\ & & & & H_{1|p-1} & H_{2|p} \end{pmatrix}\right\} (p-1) \text{ blocks}$$

(Observe that first block corresponding to $(H_{1|1}\ H_{2|2})$ reduces to (1 10), and that corresponding to $(H_{1|2}\ H_{2|3})$ reduces to $\begin{pmatrix} 10 & 100 \\ 01 & 110 \end{pmatrix}$.)

Furthermore, since by assumption the following sub-matrix has full rank:

$$\mathcal{H}2 = \left.\begin{pmatrix} H_2 & & & \\ H_1 & H_2 & & \\ \ddots & \ddots & & \\ & & H_1 & H_2 \\ & & & H_1 \end{pmatrix}\right\} (n+1) \text{ blocks}$$

the matrix $\mathcal{H}$ has full rank. Note here that in the top $48-p$ rows, $H_{1|p}$ is entirely zero. However these rows in $\mathcal{H}$ are independent since $\mathcal{H}2$ has full rank. In the

remaining rows $H_{1|p}$ is in echelon form and hence independent. Note that it has number of rows i.e., constraints:

$$48 \times (n+1) + \sum_{i=1}^{p-1} i = 48(n+1) + \frac{p(p-1)}{2}.$$

Also, note the number of variables i.e., columns is

$$64 \times n + \sum_{i=1}^{p} i = 64 \cdot n + \frac{p(p+1)}{2}.$$

Thus the nullity of the system is

$$64 \cdot n + \frac{p(p+1)}{2} - \left(48(n+1) + \frac{p(p-1)}{2}\right) = p + 16 \cdot n - 48.$$

This completes the proof.

■

Bibliography

[AGHP92] N. Alon, O. Goldreich, J. Håstad, and R. Peralta. Simple Construction of Almost k-wise Independent Random Variables. *Random Structures and Algorithms*, 3(3):289–304, 1992.

[AHRS99] Y. Aumann, J. Håstad, M. Rabin, and M. Sudan. On Linear Consistency Testing. In *Random*, 1999.

[AJK98] E. F. Assumus Jr. and J. D. Key. *Polynomial codes and Finite Geometries in Handbook of Coding Theory, Vol II , Edited by V. S. Pless Jr., and W. C. Huffman*, chapter 16. Elsevier, 1998.

[AKK+03] N. Alon, T. Kaufman, M. Krivelevich, S. Litsyn, and D. Ron. Testing Low-Degree Ploynomials over GF(2). In *Proc. of RANDOM 03*, 2003.

[AKNS99] N. Alon, M. Krivelevich, I. Newman, and M. Szegedy. Regular languages are testable with a constant number of queries. In *Proc. of Fortieth Annual Symposium on Foundations of Computer Science*, pages 645–655, 1999.

[ALM+92] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof Verification and the intractability of approximation problems. In *Proc. of IEEE Symposium of the Foundation of Computer Science*, pages 14–23, 1992.

[ALM+98] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof Verification and the intractability of approximation problems. *Journal of the ACM*, 45(3):501–555, 1998.

[AS92] S. Arora and S. Safra. Probabilistic chekcing of proofs: A new characterization of np. In *Proc. of IEEE Symposium of the Foundation of Computer Science*, pages 2–13, 1992.

[AS98] S. Arora and S. Safra. Probabilistic chekcing of proofs: A new characterization of NP. *Journal of the ACM*, 45(1):70–122, 1998.

[BC04a] E. Biham and R. Chen. Near collisions of SHA-0. In *Crypto, Lecture Notes in Computer Science 3152*, 2004.

[BC04b] E. Biham and R. Chen. New results on SHA-0 and SHA-1. In *Short talk presented at CRYPTO'04 Rump Session*, 2004.

[BCHS95] M. Bellare, D. Coppersmith, M. Håstad, J.and Kiwi, and M. Sudan. Linear testing in characteristic two. In *Proc. of Foundations of Computer Science*, pages 432–441, 1995.

[BFL91] L. Babai, L. Fortnow, and C. Lund. Non-deterministic exponential time has two prover interactive protocols. In *Computational Complexity*, pages 3–40, 1991.

[BFLS91] L. Babai, L. Fortnow, L. Levin, and M. Szegedy. Checking computations in polylogarithmic time. In *Proc. of Symposium on the Theory of Computing*, pages 21–31, 1991.

[BGS98] M. Bellare, O. Goldreich, and M. Sudan. Free bits, PCPs, and nonapproximability- towards tight results. *SIAM Journal on Computing*, 27(3):804–915, 1998.

[BLB05] S. Ballet and D. Le Brigand. On the existence of non-special divisors of degree g and $g-1$ in algebraic function fields over $\mathbb{F}_q$. In *Journal of Number Theory*, 2005. Also arXiv:math.NT/0410193.

[BLR93] M. Blum, M. Luby, and R. Rubinfeld. Self-testing/correcting with applications to numerical problems. *Journal of Computer and System Sciences*, 47:549–595, 1993.

[BOCLR04] M. Ben-Or, D. Coppersmith, M. Luby, and R. Rubinfeld. Non-Abelian Homomorphism Testing, and Distributions Close to their Self-Convolutions. In *Random*, 2004.

[BSHR03] E. Ben-Sasson, P. Harsha, and S. Raskhodnikova. Some 3CNF properties are hard to test. In *Proc. of Symposium on the Theory of Computing*, pages 345–354, 2003.

[BSS03] L. Babai, A. Shpilka, and D. Stefankovic. Locally testable cyclic codes. In *Proc. of IEEE Symposium of the Foundation of Computer Science*, pages 116–125, 2003.

[BSS05] E. Ben-Sasson and M. Sudan. Simple PCPs with polylog rate and query complexity. In *Proc. of Symposium on the Theory of Computing*, 2005.

[BSSVW03] E. Ben-Sasson, M. Sudan, S. Vadhan, and A. Wigderson. Derandomizing low degree tests via epsilon-biased spaces. In *Proc. of Symposium on the Theory of Computing*, pages 612–621, 2003.

[Che92] V. V. Chepyzhov. New lower bounds for minimum distance of linear quasi-cyclic and almost linear cyclic codes. In *Problems of information Transmission, Vol. 28, No. 1*, 1992.

[CJ98] F. Chabaud and A. Joux. Differential collisions in SHA-0. In *Crypto, Lecture Notes in Computer Science 1462*, 1998.

[CS03] D. Coppersmith and M. Sudan. Reconstructing curves in three (and higher) dimensional spaces from noisy data. In *Proc. of Annual Symposium on Theory of Computing*, 2003.

[Dam89] I Damgård. A design principle for hash functions. In *Crypto*, pages 416–427, 1989.

[DGM70] P. Delsarte, J. M. Goethals, and F. J. MacWillams. On generalized reed-muller codes and their relatives. *Information and Control*, 16:403–442, 1970.

[Din06] I. Dinur. The PCP theorem by gap amplification. In *Proc. of Symposium on the Theory of Computing*, 2006. Also available at `http://eccc.uni-trier.de/eccc-reports/2005/TR04-046/index.html/`.

[DK00] P. Ding and J. D. Key. Minimum-weight codewords as generators of generalized reed-muller codes. *IEEE Trans. on Information Theory.*, 46:2152–2158, 2000.

[DMS03] I. Dumer, D. Micciancio, and M. Sudan. Hardness of approximating the minimum distance of a linear code. In *IEEE Transaction on Information Theory, 49(1)*, 2003.

[EGH+04] Ben-Sasson E., O. Goldreich, P. Harsha, M. Sudan, and S. Vadhan. Robust PCPs of Proximity, shorter PCPs and application to coding. In *Proc. of Symposium on the Theory of Computing*, pages 1–10, 2004.

[Eli57] P. Elias. List decoding for noisy channels. Tech. Report 335, Research Lab. Electronics, MIT, 1957.

[Eli91] P. Elias. Error-correcting codes for list decoding. In *IEEE Transactions on Information Theory*, pages 37:5–12, 1991.

[FGL+91] U. Fiege, S. Goldwasser, L. Lovasz, S. Safra, and M. Szegedy. Approximating clique is almost np-complete. In *Proc. of IEEE Symposium of the Foundation of Computer Science*, pages 2–12, 1991.

[Fri86] J. Friedman. Constructing $O(n \log n)$ Size Monotone Formulae for the k-th Elementary Symmetric Polynomial of n Boolean Variables. *SIAM Journal of Computing*, 15(3):641–654, 1986.

[FS95] K. Friedl and M. Sudan. Some improvements to total degree tests. In *Proceedings of the 3rd Annual Israel symposium on Theory of Computing and Systems*, pages 190–198, 1995. Corrected version available at http://theory.lcs.mit.edu/∼madhu/papers/friedl.ps.

[GI01] V. Guruswami and P. Indyk. Expander-based constructions of efficiently decodable codes. In *Proceedings of the 42nd IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 658–667, 2001.

[GI03] V. Guruswami and P. Indyk. Linear-time encodable and list decodable codes. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing (STOC)*, pages 126–135, June 2003.

[GJ79] M. R. Garey and D. S. Johnson. *Computers and Intractability: A guide to the Theory of* NP*-completeness*. W. H. Freeman, 1979.

[Gol49] M. J. E. Golay. Notes on digital coding. *Poc. of IEEE.*, 37:657, 1949.

[Gol05] O. Goldreich. Short Locally Testable Codes and Proofs (Survey). Electronic Colloquium on Computational Complexity (ECCC)(014), 2005. http://eccc.uni-trier.de/eccc-reports/2005/TR04-014/index.html/.

[Gop81] V. D. Goppa. Codes on algebraic curves. In *Soviet Math. Doklady*, pages 24:170–172, 1981.

[Gow98] W. T. Gowers. A new proof of Szemeredi's theorem for arithmetic progressions of length four. *GAFA (Geometric and Functional Analysis)*, 8(1):529–551, 1998.

[Gow01] W. T. Gowers. A new proof of Szemeredi's theorem. *GAFA (Geometric and Functional Analysis)*, 11(3):465–588, 2001.

[GP07] V. Guruswami and A. C. Patthak. Correlated algebraic-geometric codes: improved list decoding over bounded alphabets. *Mathematics of Computation, To appear. Preliminary version appeared in FOCS'06* , 2007.

[GR06] V. Guruswami and A. Rudra. Explicit capacity-achieving list-decodable codes. In *Proceedings of the 38th ACM Symposium on Theory of Computing*, pages 1–10, May 2006.

[GS95a] A. Garcia and H. Stichtenoth. A tower of Artin-Schreier extensions of function fields attaining the Drinfeld-Vladut bound. In *Invent. Math.*, 1995.

[GS95b] A. Garcia and H. Stichtenoth. A tower of Artin-Schreier extensions of function fields attaining the Drinfeld-Vlădut bound. *Inventiones Mathematicae*, 121:211–222, 1995.

[GS96a] A. Garcia and H. Stichtenoth. On the asymptotic behavior of some towers of function fields over finite fields. *Journal of Number Theory*, 61(2):248–273, 1996.

[GS96b] A. Garcia and H. Stichtenoth. On the asymptotic behaviour of some towers of function fields over finite fields. In *Journal of Number Theory*, 1996.

[GS99] V. Guruswami and M. Sudan. Improved decoding of Reed-Solomon codes and algebraic-geometry codes. In *IEEE Transactions on Information Theory*, pages 45(6): 1757–1767, 1999.

[GS01] V. Guruswami and M. Sudan. On representations of algebraic-geometric codes. *IEEE Transactions on Information Theory*, 47(4):1610–1613, May 2001.

[GS02] O. Goldreich and M. Sudan. Locally Testable Codes and PCPs of almost linear length. In *Proc. of Symposium on Foundations of Computer Science*, pages 13–22, 2002.

[GT05] B. Gren and T. Tao. An inverse theorem for the Gowers U^3 norm. *math.NT/0503014*, 2005.

[Gur01] V. Guruswami. *List-decoding of Error Correcting Codes*. PhD thesis, MIT, 2001. "Also Lecture notes in Computer Science, 3282, Springer, New York 2005.".

[Ham50] R. W. Hamming. Error Detecting and Error Correcting Codes. *Bell System Technical Journal*, 29:147–160, April 1950.

[Har77] R. Hartshorne. *Algebraic Geometry*. Springer-Verlag, 1977.

[IT03] K. Iwama and S. Tamaki. Improved upper bounds for 3-SAT. Electronic Colloquium on Computational Complexity (ECCC)(053), 2003. `http://eccc.uni-trier.de/eccc-reports/2003/TR03-053/index.html/`.

[JP05a] C. S. Jutla and A. C. Patthak. A Matching Lower Bound on the Minimum Weight of SHA-1 Expansion Code. Cryptology ePrint Archive, Report 2005/266, 2005. `http://eprint.iacr.org/`.

[JP05b] C. S. Jutla and A. C. Patthak. A Simple and Provably Good Code for SHA Message Expansion. NIST CRYPTOGRAPHIC HASH WORKSHOP, 2005 Also available at Cryptology ePrint Archive, Report 2005/247, 2005. `http://eprint.iacr.org/`.

[JP05c] C. S. Jutla and A. C. Patthak. Is SHA-1 conceptually sound? Cryptology ePrint Archive, Report 2005/350, 2005. `http://eprint.iacr.org/`.

[JP06] C. S. Jutla and A. C. Patthak. Provably Good Code for Hash Function Design. Selected Areas in Cryptography, 2006.

[JPR04] C. S. Jutla, A. C. Patthak, and A. Rudra. Testing polynomials over general fields. manuscript, 2004.

[JPRZ04] C. S. Jutla, A. C. Patthak, A. Rudra, and D. Zuckerman. Testing low-degree polynomials over prime fields. In *Proc. of IEEE Symposium of the Foundation of Computer Science*, 2004.

[KAK+01] Shum. K., I. Aleshnikov, P. V. Kumar, H. Stichtenoth, and V. Deolalikar. A low-complexity algorithm for the construction of algebraic-geometric codes better than the Gilbert-Varshamov bound. *IEEE Transactions on Information Theory*, 47(6):2225–2241, 2001.

[KL05] T. Kaufman and S. Litsyn. Almost orthogonal linear codes are locally testable. In *To appear in Proc. of IEEE Symposium of the Foundation of Computer Science*, 2005.

[KLP68] T. Kasami, S. Lin, and W. W. Peterson. New Generalization of the Reed-Muller Codes Part I: Primitive Codes. *IEEE Transactions on Information Theory* , IT-14(2):189–199, March 1968.

[KR04] T. Kaufman and D. Ron. Testing polynomials over general fields. In *Proc. of IEEE Symposium of the Foundation of Computer Science*, 2004.

[KV03] R. Koetter and A. Vardy. Algebraic soft-decision decoding of Reed-Solomon codes. *IEEE Transactions on Information Theory*, 49(11):2809–2825, 2003.

[Lal03] K. Lally. Quasicyclic codes of index ℓ over $\mathbb{F}_q$ Viewed as $\mathbb{F}_q[x]$-submodules of $\mathbb{F}_{q^l}[x]/\langle x^m - 1\rangle$. In *Lecture Notes in Computer Science, Vol. 2643, Springer*, 2003.

[LN94] R. Lidl and H. Niederreiter. *Introduction to Finite Fields and their Applications*. Cambridge University Press, 1994.

[LS05] S. Ling and P. Solé. Structure of quasi-clcyic codes III: Generator theory. In *IEEE Transaction on Information Theory*, 2005.

[Mer89] R. C. Merkle. One way hash functions and DES. In *Crypto*, pages 428–446, 1989.

[MP05] K. Matusiewicz and J. Pieprzyk. Finding good differential patterns for attacks on SHA-1. In *International Workshop on Coding and Cryptography*, 2005.

[MS77] F. J. MacWilliams and N. J. A. Sloane. *The Theory of Error-Correcting Codes*. North-Holland Mathematical Library, 1977.

[Pat07] A. Patthak. Conditional inverse theorem. *Manuscript*, April 2007.

[PV05] F. Parvaresh and A. Vardy. Correcting errors beyond the Guruswami-Sudan radius in polynomial time. In *Proceedings of the 46th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 285–294, 2005.

[Riv92] R. Rivest. RFC1321: The MD5 message-digest algorithm. In *Internet Activities Board*, 1992.

[RO05] V. Rijmen and E. Oswald. Update on SHA-1. In *Lecture Notes in Computer Science, Vol. 3376, Springer*, 2005.

[RS96] R. Rubinfeld and M. Sudan. Robust characterizations of polynomials with applications to program testing. *SIAM Journal on Computing* , 25(2):252–271, 1996.

[Sam07] A. Samorodnitsky. Low degree tests at large distance. In *STOC*, 2007. To appear.

[Sch99] U. Schoning. A probablistic algorithm for k-SAT and constriant satisfaction problems. In *Proc. of Foundations of Computer Science*, 1999.

[Sha48] C. E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423, 623–656, 1948.

[Sha79] A. Shamir. How to share a secret. *Communications of the ACM*, 22:612–613, 1979.

[ST06] A. Samorodnitsky and L. Trevisan. Gowers Uniformity, Influence of Variables, and PCPs. In *STOC*, 2006.

[Sti91] H. Stichtenoth. *Algebraic Function Fields and Codes*. Springer, 1991.

[Sti93] H. Stichtenoth. *Algebraic Function Fields and Codes*. Universitext, Springer-Verlag, Berlin, 1993.

[STV01] M. Sudan, L. Trevisan, and S. Vadhan. Pseudorandom Generators without the XOR lemma. *Journal of Computer and System Sciences*, 62(2):236–266, 2001.

[SU01] R. Shaltiel and C. Umans. Simple Extractors for All Min-Entropies and a New Pseudo-Random Generator. In *Proc. of Foundations of Computer Science*, pages 648–657, 2001.

[Sud97] M. Sudan. Decoding of Reed-Solomon codes beyond the error-correction bound. In *Journal of Complexity*, pages 12: 180–193, 1997.

[Sud01] M. Sudan. Lecture notes on algorithmic introduction to coding theory, Fall 2001. Lecture 15.

[SW98] M. A. Shakrollahi and H. Wasserman. Decoding algebraic-geometric codes beyond the error-correction bound. In *Proc. of Annual Symposium on Theory of Computing*, 1998.

[Tre01] L. Trevisan. Extractors and pseudorandom generators. *Journal of the ACM*, 48(4):860–879, 2001.

[TVZ82] M. A. Tsfasman, S. G. Vladut, and T. Zink. Modular Curves, Shimura Curves and codes better than the Varshamov-Gilbert bound. In *Math. Nachrichten*, pages 109: 21–28, 1982.

[TW67] R. L. Townsend and E. J. Weldon. Self-orthogonal quasi-cyclic codes. In *IEEE Transaction on Information Theory*, 1967.

[TZ04] A. Ta-Shma and D. Zuckerman. Extractor codes. *IEEE Transactions on Information Theory*, 50(12):3015–3025, 2004.

[TZS01] A. Ta-Shma, D. Zuckerman, and S. Safra. Extractors from Reed-Muller Codes. In *Proc. of Foundations of Computer Science*, 2001.

[Uni93] United States Department of Commerce, National Institute of Standards and Technology, Federal Information Processing Standard Publication #180. *Secure Hash Standard*, 1993.

[Uni95] United States Department of Commerce, National Institute of Standards and Technology, Federal Information Processing Standard Publication #180-1 (addendum to [Uni93]). *Secure Hash Standard*, 1995.

[Uni02] United States Department of Commerce, National Institute of Standards and Technology, Federal Information Processing Standard Publication #180-2. *Secure Hash Standard*, August, 2002.

[Var97] A. Vardy. The intractability of computing the minimum distance of a code. In *IEEE Transaction on Information Theory, 43(6)*, 1997.

[vL98] J. H. van Lint. *Introduction to Coding Theory*. Springer, 1998.

[vN56] J. von Neumann. Probabilistic logics and the synthesis of reliable organisms from unreliable components. *C. E. Shannon and J. McCarthy, editors, Automata Studies*, pages 43–98, 1956.

[VW07] E. Viola and A. Wigderson. Norms, XOR lemmas and lower bounds for GF(2) polynomials and multiparty protocols. In *Computational Complexity*, 2007. To appear.

[Wan97a] X. Y. Wang. The collision attack on SHA-0. In Chinese, 1997.

[Wan97b] X. Y. Wang. The Improved collision attack on SHA-0. In Chinese, 1997. `http://www.infosec.edu.cn/`.

[Woz58] M. J. Wozencraft. List decoding. Quarterly Progress Report, Research Lab. Electronics, MIT, 48:90-95, 1958.

[WYY05a] X. Wang, A. Yao, and F. Yao. New collision search for SHA-1. In *Short talk presented at CRYPTO'05 Rump Session*, 2005.

[WYY05b] X. Wang, H. Yu, and Y. L. Yin. Efficient collision search attacks in SHA-0. In *Crypto, Lecture Notes in Computer Science 3621*, 2005.

[WYY05c] X. Wang, H. Yu, and Y. L. Yin. Finding collisions in the full SHA-1. In *Crypto, Lecture Notes in Computer Science 3621*, 2005.

Vita

Anindya Chandra Patthak was born in Uttarpara, India on January 1st, 1977, the son of Swapna Patthak and Gopal C. Patthak. After completing his work at Uttarpara Govt. High School, India in 1995, he attended Indian Institute of Technology, Kharagpur, India. He received the degree of Bachelor of Technology, Hons. in Computer Sc. & Engg in 1999 frim IIT Kharagpur, India. After completing his graduation, he spent a year at his alma mater doing research. He spent a year at Duke University starting in August 2000. In August 2001 he entered the Graduate School of the University of Texas at Austin.

Permanent Address: 1520 N Beckley Ave, APT 935, Dallas, TX 75203

This dissertation was typeset with LaTeX 2_ε[1] by the author.

[1] LaTeX 2_ε is an extension of LaTeX. LaTeX is a collection of macros for TeX. TeX is a trademark of the American Mathematical Society. The macros used in formatting this dissertation were written by Dinesh Das, Department of Computer Sciences, The University of Texas at Austin, and extended by Bert Kay, James A. Bednar, and Ayman El-Khashab.