EXTRACTORS FOR ADDITIVE STRUCTURES AND SPACE-BOUNDED COMPUTATION

A Dissertation

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

by

Jyun-Jie Liao

May 2024

Extractors for Additive Structures and Space-bounded Computation

Jyun-Jie Liao, Ph.D.

Cornell University 2024

Randomness is a powerful resource in computer science, with rich applications in algorithm design, cryptography, distributed computing, etc. Most of the applications assume access to a sequence of uniform and independent random bits. However, the randomness we collect from nature (e.g., activity of CPU, atmospheric noise, radioactive decay) does not seem as perfect. This motivates the study of *randomness extractors*, which are deterministic algorithms that can convert an imperfect random source (with some entropy) into a uniform random string. The ultimate goal in the area of randomness extraction is to construct an extractor that can work for any source that we would ever see in nature and applications. Unfortunately, a folklore result shows that it is impossible to construct an extractor that can work for every source that has entropy. It is therefore necessary to assume that the given source has certain structure, but we still hope that the structure we assume is as general as possible.

In this thesis, we first focus on the construction of randomness extractors for sources with *additive structure*. This includes *affine sources*, which are uniform distributions over affine subspaces; and more generally *the sum of two independent sources*, which is a surprisingly general model that contains many other natural sources such as independent sources, affine sources and sources samplable by space-bounded computation. For both models, we construct extractors that improve the previous state-of-the-art, and develop many useful tools along the way. In addition, we discover new connections between sources with additive structure and space-bounded

computation, and as a result we obtain extractors for small-space sources with optimal entropy requirement, and a new lower bound for linear branching programs. Finally, we develop more results regarding randomness and space-bounded computation, including new weighted pseudorandom generators and derandomization bounds for small-space algorithms.

BIOGRAPHICAL SKETCH

Jyun-Jie grew up in Taichung, Taiwan. In 2016, he got his Bachelor's degree in Electrical Engineering and Computer Science from National Chiao Tung University (now known as National Yang Ming Chiao Tung University). Afterward, he worked as a research assistant at Academia Sinica under the supervision of Dr. Kai-Min Chung. In 2018, he joined the Computer Science department at Cornell Univesity as a Ph.D. student, advised by Prof. Eshan Chattopadhyay.

ACKNOWLEDGEMENTS

I want to express my deepest appreciation to the guidance provided by my advisor, Eshan Chattopadhyay. Eshan has always been very supportive and generous with his time. He kindly taught me everything about research and patiently listened to all my naive and unorganized ideas in my initial years. In addition, he always shares many invaluable insights on research with me, helps me improve a lot in writing, and provides me with many great suggestions about academic life. I have learned so much from Eshan and am still learning more from him.

I also want to thank my special committee members, Bobby Kleinberg and Elaine Shi. Bobby has always been a reliable source of knowledge in mathematics and theoretical computer science. His lectures, talks and even casual chats about math have been very insightful to me. Elaine kindly guided me through various fascinating topics in cryptography in my first year. Without her help, the start of my PhD journey wouldn't have been as smooth and enjoyable. Also, I want to thank all the faculty in Cornell theory group for building such an inclusive environment. A special thanks to Noah Stephens-Davidowitz for sharing his experience when I was panicking about my career.

Furthermore, I want to thank Kai-Min Chung for introducing me to theoretical computer science. The time I spent in Academia Sinica under his supervision helped me build a solid background in research. I wouldn't even know what a randomness extractor is if Kai-Min hadn't hosted the extractor seminar.

I have been lucky to work with many amazing coauthors during my PhD, including Marshall Ball, Eshan Chattopadhyay, Jesse Goodman, Tal Malkin and Li-Yang Tan. It wouldn't be possible to complete the projects I have worked on without them, and I learned a lot from collaborating with them. It's also a pleasure to have been hosted by Mahdi Cheraghchi and Li-Yang Tan during my summer visits. They

shared many interesting ideas with me, and guided me to think about research in different perspectives. Moreover, I want to thank Kaave Hosseini for all the time we have spent collaborating and his many interesting ideas on additive combinatorics and related topics. I also want to thank all the amazing researchers who have shared helpful comments on papers in this thesis. An incomplete list includes: William Hoza, Xin Lyu, Ted Pyne, Salil Vadhan, Hongxun Wu, David Zuckerman.

During my six years in Ithaca, I have been supported physically and mentally by many friends, including the extraction team: Jesse, Mohit, Nomi, Yunya; other Gates 336 folks: Shijin, Makis, Sloan, Spencer, Giannis, Surendra; my dinner buddies: Wen-Ding, Yu-Ju; and a short and incomplete list of many other friends who have been in the CS department: Wei-Kai, Haobin, Siqiu, Raunak, Jason, Oli, Ayush, Abhishek, Princewell, and many others. A special shoutout to my extractor bro, Jesse Goodman, for all his support on research and life. Now that I am also leaving the theory lab, I will miss the time we spent grinding together at 3 am.

Finally, I want to thank my family for supporting me in pursuing this degree. It's been a long journey, and I am almost there.

TABLE OF CONTENTS

Chapter 1

Introduction

Randomness is a powerful tool in computer science, and has been widely used in areas such as algorithm design, cryptography, distributed computing, etc. Most of the applications assume access to perfect randomness, i.e., a stream of uniform and independent random bits. However, natural sources of randomness often generate biased and correlated random bits, and in cryptographic applications there are many scenarios where some information about the random bits we use is leaked to the adversary. This motivates the area of randomness extraction, which aims to construct *randomness extractors*, which are deterministic algorithms that convert an imperfect random source into an almost uniform random string.

Formally, the amount of randomness in an imperfect random source $\mathbf{X}$ is captured by its *min-entropy*.

Definition 1.1. *The min-entropy of a distribution* $\mathbf{X}$ *is*

$$\mathrm{H}_\infty(\mathbf{X}) = \min_{x \in \mathrm{Supp}(\mathbf{X})} (-\log(\Pr[\mathbf{X} = x])).^{1}$$

We call $\mathbf{X} \in \{0,1\}^n$ *a* (n,k)*-source if it satisfies* $\mathrm{H}_\infty(\mathbf{X}) \geq k$*.*

[1] We use $\log$ to denote the base-2 logarithm through out this thesis.

Ideally we want a deterministic function Ext with entropy requirement $k \ll n$, i.e., for every (n,k)-source $\mathbf{X}$ the output $\mathrm{Ext}(\mathbf{X})$ is close to a uniform string. Unfortunately, a folklore result shows that it is impossible to construct such a function even when $k = n-1$: for a fixed extractor Ext with 1-bit output, the uniform distribution over the pre-image of either 0 or 1 will be a source with min-entropy $\geq n-1$.

To bypass the impossibility result, researchers have explored two different approaches. The first one is based on the notion of *seeded extraction*, introduced by Nisan and Zuckerman [NZ96]. This approach assumes that the extractor has access to a short independent uniform random seed, and the extractor needs to convert the given source $\mathbf{X}$ into a uniform string. Through a successful line of research we now have seeded extractors with almost optimal parameters [LRVW03, GUV09, DKSS13]. The second approach is *deterministic extraction*, which assumes some structure in the given source. Formally, a deterministic extractor is defined as follows.

Definition 1.2. *Let $\mathcal{X}$ be a family of distribution over $\{0,1\}^n$. We say a deterministic function $\mathrm{Ext} : \{0,1\}^n \to \{0,1\}^m$ is a deterministic extractor for $\mathcal{X}$ with error ε if for every distribution $\mathbf{X} \in \mathcal{X}$, $\mathrm{Ext}(\mathbf{X})$ is ε-close to uniform in statistical distance.*[2] *We say Ext is explicit if Ext is computable by a polynomial-time algorithm.*

Ideally, we want the error ε to be small, the output length m to be long, and the class of sources $\mathcal{X}$ to be as general as possible. In addition, $\mathcal{X}$ typically contains an entropy requirement condition (along with the structure assumption), i.e., every source $\mathbf{X}$ in $\mathcal{X}$ satisfies $\mathrm{H}_\infty(\mathbf{X}) \geq k$ for some k, and we want k to be as low as possible.

The most well-studied deterministic extractors are multi-source extractors, which assume that the extractor is given C independent (n,k)-sources

[2] See Chapter 2 for the definition of statistical distance.

$\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_C$. This model was first introduced by Chor and Goldreich [CG88]. They constructed explicit two-source extractors with error $2^{-\Omega(n)}$ for entropy $0.51n$, and proved that there exists a two-source extractor for entropy $k = O(\log(n))$ with error $2^{-\Omega(k)}$. Significant progress was made by Chattopadhyay and Zuckerman [CZ19], who showed how to construct an extractor for two sources with entropy $k = \text{polylog}(n)$, after a long line of works on independent source extractors (see the references in [CZ19]). The output length was later improved to $\Omega(k)$ by Li [Li16]. Furthermore, Ben-Aroya, Doron and Ta-Shma [BDT19] showed how to improve the entropy requirement to $O(\log^{1+o(1)}(n))$ for constant error and 1-bit output. The entropy requirement was further improved in subsequent works [Coh17, Li17, Li19], and was eventually improved to asymptotically optimal ($k = O(\log(n))$) in a very recent work by Li [Li23].

Apart from independent sources, many other classes of sources have been studied for deterministic extraction. We briefly introduce some of these research directions that are closely related to this thesis. A well-studied class is oblivious bit-fixing sources [CGH+85, GRS06, KZ07, Rao09], where some unknown coordinates are uniform and independent and the remaining coordinates are fixed. Extractors for such sources have found applications in cryptography [CGH+85, KZ07]. A natural generalization of bit-fixing sources is the class of affine sources, which are uniform distributions over some affine subspaces of dimension k in a field $\mathbb{F}_q^n$. For the setting of large field size ($q = \text{poly}(n)$), Gabizon and Raz [GR07] constructed an affine extractor that can even work for lines (i.e., $k = 1$). More generally, they showed how to extract most of the entropy out of any affine source in this large field setting. A construction with improved error was given by Bourgain, Dvir, and Leeman [BDL16] assuming q is a prime, and further that $q - 1$ does not have too many prime factors. The task of constructing affine extractors is generally more challenging as the field

size gets smaller. In the setting of $q = O(1)$, Bourgain [Bou07] and subsequent works of Yehudayoff [Yeh11] and Li [Li11] gave explicit constructions for $k \geq n/\sqrt{\log \log n}$. DeVos and Gabizon [DG10] obtained a trade-off between the field size and entropy, and gave explicit affine extractors for fields with characteristic $\Omega(n/k)$ and size $q = \Omega((n/k)^2)$. Thus, they require linear entropy (i.e., $k = \Omega(n)$) for extraction from affine sources on fields with constant characteristic. A vastly improved result was obtained by Li [Li16] who gave an explicit affine extractor that works for $q = 2$ (which is generally considered the hardest setting) and $k \geq C(\log n)^C$, for some large enough constant C.

Another important line of work focused on the class of samplable sources, which are sources sampled by "low-complexity procedures" such as efficient algorithms [TV00, BGDM23], small-space algorithms [KRVZ11] or constant-depth circuits [Vio14]. Researchers have also studied interleaved sources [RY11, CZ16, CL16b, CL20], which is a generalization of independent sources such that the bits from different independent sources are permuted in an unknown order.

1.1 Our Contributions

In this thesis, we mainly focus on sources with additive structure. More precisely, we study extractors for *sumset sources*, a model that was first introduced by Chattopadhyay and Li [CL16b]. (Here we view $\{0,1\}^n$ and $\mathbb{F}_2^n$ as equivalent. Therefore "sum" also means "bit-wise XOR" for strings.)

Definition 1.3. *We say a source $\mathbf{X} \in \mathbb{F}_2^n$ is a C-*sumset source *with min-entropy k if there exist C independent (n,k)-sources $\{\mathbf{X}_i\}_{i\in[C]}$ such that $\mathbf{X} = \sum_{i=1}^{C} \mathbf{X}_i$.*

The name "sumset" comes from the notion of "sum of sets", which is a central concept that is studied in additive combinatorics [TV06].

Definition 1.4 (sumsets)**.** *For any sets $A, B \subseteq \mathbb{F}_2^n$, the sumset $A + B$ is defined as $\{a + b : a \in A, b \in B\}$.*

Our first main result in this thesis is an extractor for *affine sources* (over $\mathbb{F}_2^n$) with min-entropy $\tilde{O}(\log(n))$. This beats the prior state-of-the-art by Li [Li16] that requires min-entropy $\mathrm{polylog}(n)$. (This result is from a joint work with Chattopadhyay and Goodman [CGL21].) Note that an affine source can be viewed as a sumset source with "maximum additive structure" because one can write an affine source $\mathbf{X}$ as the sum of infinitely many independent copies of $\mathbf{X}$ and a constant.

Definition 1.5. *We say $\mathbf{X}$ is an (n, k)-affine source if $\mathbf{X}$ is uniform over an affine subspace of dimension k.*[3]

Theorem 1.6. *For any constant $\varepsilon > 0$, there is an extractor for $(n, \tilde{O}(\log(n)))$-affine source with error ε.*

Our second main result is an extractor for 2-sumset sources with min-entropy $\mathrm{polylog}(n)$. We can further improve the entropy requirement to $\tilde{O}(\log(n))$ at the cost of getting a larger error (with the help of some ideas we developed in [CGL21]). (These results are from a joint work with Chattopadhyay [CL22].)

Theorem 1.7. *There is an extractor with error $n^{-\Omega(1)}$ for n-bit 2-sumset sources with min-entropy $\mathrm{polylog}(n)$.*

Theorem 1.8. *For any constant $\varepsilon > 0$, there is an extractor with error ε for n-bit 2-sumset source with min-entropy $\tilde{O}(\log(n))$.*

[3] Note that the min-entropy of an (n, k)-affine source is exactly k.

We note that 2-sumset sources is a very general class which captures independent sources, interleaved sources, affine sources and small-space sources, so our explicit extractors have many nice applications. In addition, our result is a significant improvement over prior works because it wasn't even known whether there exists a (non-explicit) extractor for 2-sumset extractor with $o(n)$ entropy before our work. (See Chapter 5 for more details.)

The third main result is a new reduction from extractor for small-space samplable sources to sumset extractors. The formal definition of small-space samplable sources can be found in Chapter 5. (This result is also from [CL22].)

Lemma 1.9. *Any extractor with error ε for n-bit 2-sumset source with min-entropy k is also an extractor with error $O(\varepsilon)$ for sources with min-entropy $2s + 2k + 2\log(n/\varepsilon)$ that are samplable in space s.*

Notably, a result by Kamp, Rao, Vadhan and Zuckerman [KRVZ11] shows that any extractor for sources samplable in space s must have min-entropy requirement at least $2s$. Therefore, with our explicit extractors for 2-sumset sources we obtain explicit extractor for small-space source with *optimal* dependence on the space parameter.

The fourth main result is an application of sumset extractor for circuit lower bounds, in a model called read-once linear branching programs [GPT22]. Roughly speaking, a linear branching program is a non-uniform small-space computation process that can make arbitrarily linear queries to the input, and the read-once property is an additional restriction that ensures that all the linear queries that have been made are linearly independent. We leave the formal definition to Chapter 6, and here we only point out the fact that any function can be trivially computed with a read-once linear branching program of size $2^{n-\log(n)}$. We show that a sumset ex-

tractor with low entropy requirement cannot be computed with a read-once linear branching program with size much better than the trivial bound. (This result is from a joint work with Chattopadhyay [CL23]).

Theorem 1.10. *Let* Ext *be an extractor for* 2*-sumset sources with min-entropy* k*. Then* Ext *cannot be computed by a read-once linear branching program with size less than* $2^{n-2k-O(1)}$.

Finally, we show a few more results on derandomization of space-bounded computation. These results are not directly related to deterministic extractors, but they share the same theme with randomness extraction: we know that randomness is useful, but what if perfect randomness is expensive? The space-bounded derandomization problem considers the case that we have a randomized algorithm with space complexity s, and asks whether we can completely remove the need of randomness while not increasing the space complexity by too much.

Specifically, recall that $\mathbf{BPL}$ represents the class of decision problems that can be computed by a $O(\log(n))$-space randomized algorithm that always halts. It is widely conjectured that we can compute any problem in $\mathbf{BPL}$ deterministically in space $O(\log(n))$, i.e., $\mathbf{BPL} = \mathbf{L}$. However, the best-known bound for this conjecture before 2021 was $\mathbf{BPL} \subseteq \mathbf{L}^{1.5}$, which was proved by Saks and Zhou [SZ99] in the 90s.[4] In 2018, Braverman, Cohen and Garg [BCG20] introduced the surprisingly powerful notion of weighted pseudorandom generators (WPRGs). Their work initiated a line of research that culminated in Hoza's new bound for derandomization of $\mathbf{BPL}$: $\mathbf{BPL} \subseteq \mathbf{L}^{1.5-o(1)}$ [Hoz21a]. In Chapter 7 and Chapter 8, we show our contributions in this line of works, including WPRG constructions for general and regular read-once branching programs, an argument on how to apply Saks-Zhou framework in WPRG and a derandomization result for regular read-once branching

[4] $\mathbf{L}^{\alpha}$ denotes the class of decision problems that can be computed deterministically in space $O(\log^{\alpha}(n))$.

programs. (These results are from joint works with Chattopadhyay [CL20, CL24].)

Chapter 2

Preliminaries

2.1 Notation

Basic notation. The logarithm in this thesis is always base-2. For every $n \in \mathbb{N}$, $[n]$ denotes $\{1, 2, \ldots, n\}$. Throughout this thesis, $\{0,1\}^n$ and $\mathbb{F}_2^n$ are interchangeable (with the natural bijection between them), and so are $\{0,1\}^n$ and $[2^n]$ (with any efficiently computable bijection). We use $x \circ y$ to denote the concatenation of two strings x and y. We say a function is explicit if it is computable by a polynomial-time algorithm, unless specified otherwise. For $x, y \in \mathbb{R}$ we use $x \approx_\varepsilon y$ to denote $|x - y| \leq \varepsilon$ and $x \not\approx_\varepsilon y$ to denote $|x - y| > \varepsilon$. For every function $f : \mathcal{X} \to \mathcal{Y}$ and set $A \subseteq \mathcal{X}$, define $f(A) = \{f(x) : x \in A\}$. For a set $A \subseteq \mathcal{X}$ we use $\mathbb{1}_A : \mathcal{X} \to \{0,1\}$ to denote the indicator function of A such that $\mathbb{1}_A(x) = 1$ if and only if $x \in A$.

Distributions and random variables. We abuse notation and treat distributions and random variables as the same. We always write a random variable/distribution in boldface font. We use $\mathrm{Supp}(\mathbf{X})$ to denote the support of a distribution. We use $\mathbf{U}_n$

to denote the uniform distribution on $\{0,1\}^n$. When $\mathbf{U}_n$ appears with other random variables in the same joint distribution, $\mathbf{U}_n$ is considered to be independent of other random variables. Sometimes we omit the subscript n of $\mathbf{U}_n$ if the length is less relevant and is clear in the context. Throughout this paper, "entropy" means min-entropy, unless specified differently.

When there is a sequence of random variables $\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_t$ in the context, for every set $S \subseteq [t]$ we use $\mathbf{X}_S$ to denote the subsequence of random variables which use indices in S as subscript, i.e., $\mathbf{X}_S := (\mathbf{X}_i)_{i \in S}$. We also use similar notation for indices on superscript. In addition, we use $\mathbf{X}_{<i}$ to denote $(\mathbf{X}_1, \ldots, \mathbf{X}_{i-1})$ and $\mathbf{X}_{\leq i}$ to denote $(\mathbf{X}_1, \ldots, \mathbf{X}_i)$. If $\mathbf{X}_0$ is defined in the context, then $\mathbf{X}_0$ is also included in $\mathbf{X}_{<i}$ and $\mathbf{X}_{\leq i}$. In the "tampering setting" (see Chapter 3 for explanation), if $\mathbf{X}^i$ is the tampered version of $\mathbf{X}$ and we define $\mathbf{R} := f(\mathbf{X})$ for some function f, then we automatically define $\mathbf{R}^i := f(\mathbf{X}^i)$ (similarly for any function that takes multiple random variables as input.)

2.2 Random Variables

2.2.1 Statistical Distance

Definition 2.1. *Let $\mathbf{D}_1, \mathbf{D}_2$ be two distributions over the same sample space Ω. The* statistical distance *between $\mathbf{D}_1$ and $\mathbf{D}_2$ is*

$$\begin{aligned}\Delta(\mathbf{D}_1; \mathbf{D}_2) &:= \max_{T \subseteq \Omega} \Big(\Pr[\mathbf{D}_1 \in T] - \Pr[\mathbf{D}_2 \in T] \Big) \\ &= \frac{1}{2} \sum_{s \in \Omega} |\mathbf{D}_1(s) - \mathbf{D}_2(s)| \,.\end{aligned}$$

We say $\mathbf{D}_1$ is ε-close to $\mathbf{D}_2$ if $\Delta(\mathbf{D}_1;\mathbf{D}_2) \leq \varepsilon$, which is also denoted by $\mathbf{D}_1 \approx_\varepsilon \mathbf{D}_2$. Specifically, when there are two joint distributions $(\mathbf{X},\mathbf{Z})$ and $(\mathbf{Y},\mathbf{Z})$ such that $(\mathbf{X},\mathbf{Z}) \approx_\varepsilon (\mathbf{Y},\mathbf{Z})$, we sometimes write $(\mathbf{X} \approx_\varepsilon \mathbf{Y}) \mid \mathbf{Z}$ for short.

We frequently use the following standard properties.

Lemma 2.2. *For every distribution $\mathbf{D}_1, \mathbf{D}_2, \mathbf{D}_3$ on the same sample space, the following properties hold:*

- *(Data processing inequality) For any distribution $\mathbf{Z}$,*

$$\Delta\left((\mathbf{D}_1,\mathbf{Z});(\mathbf{D}_2,\mathbf{Z})\right) = \mathop{\mathbb{E}}_{z\sim\mathbf{Z}}\left[\Delta\left(\mathbf{D}_1|_{\mathbf{Z}=z};\mathbf{D}_2|_{\mathbf{Z}=z}\right)\right].$$

- *For every function f, $\Delta\left(f(\mathbf{D}_1);f(\mathbf{D}_2)\right) \leq \Delta\left(\mathbf{D}_1;\mathbf{D}_2\right)$.*
- *(Triangle inequality) $\Delta\left(\mathbf{D}_1;\mathbf{D}_3\right) \leq \Delta\left(\mathbf{D}_1;\mathbf{D}_2\right) + \Delta\left(\mathbf{D}_2;\mathbf{D}_3\right)$.*

2.2.2 Bounded Independence

Definition 2.3. *A distribution $\mathbf{X} = (\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_n)$ on $\{0,1\}^n$ is called t-wise independent if for every subset $T \subseteq [n]$ of size t we have $\mathbf{X}_T = \mathbf{U}_t$.*

The following lemma analyzes the distance between "almost t-wise independent" distribution and an actual t-wise independent distribution.

Lemma 2.4 ([AGM03])**.** *Let $\mathbf{X} = (\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_n)$ be a distribution on $\{0,1\}^n$. If for every $S \subseteq [n]$ s.t. $|S| \leq t$,*

$$\bigoplus_{i\in S} \mathbf{X}_i \approx_\gamma \mathbf{U}_1,$$

then $\mathbf{X}$ is $2n^t\gamma$-close to a t-wise independent distribution.

2.2.3 Conditional Min-entropy

First we define the notion of (worst-case) conditional min-entropy.

Definition 2.5. *For a joint distribution* $(\mathbf{X}, \mathbf{Z})$*, the* conditional min-entropy *of* $\mathbf{X}$ *given* $\mathbf{Z}$ *is*

$$\mathrm{H}_\infty(\mathbf{X} \mid \mathbf{Z}) := \min_{z \in \mathrm{Supp}(\mathbf{Z})} \left(\mathrm{H}_\infty(\mathbf{X}|_{\mathbf{Z}=z})\right).$$

When conditioned on a distribution with small support we have the following lemma.

Lemma 2.6 ([MW97])**.** *Let* $\mathbf{X}, \mathbf{Z}$ *be (correlated) random variables. For every* $\varepsilon > 0$*,*

$$\Pr_{z \sim \mathbf{Z}} \left[\mathrm{H}_\infty(\mathbf{X}|_{\mathbf{Z}=z}) \geq \mathrm{H}_\infty(\mathbf{X}) - \log(\mathrm{Supp}(\mathbf{Z})) - \log(1/\varepsilon)\right] \geq 1 - \varepsilon.$$

Given the definition of conditional min-entropy, we define a block source as follows.

Definition 2.7 ([CG88])**.** $(\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_t) \in (\{0,1\}^n)^t$ *is called a* (t, n, k)*-block source if for every* $i \in [t]$ *,* $\mathrm{H}_\infty(\mathbf{X}_i \mid \mathbf{X}_1, \ldots, \mathbf{X}_{i-1}) \geq k$*. In addition, we for* $(\mathbf{X}_1, \mathbf{X}_2) \in \{0,1\}^{n_1} \times \{0,1\}^{n_2}$ *where* $\mathrm{H}_\infty(\mathbf{X}_1) \geq k_1$ *and* $\mathrm{H}_\infty(\mathbf{X}_2 \mid \mathbf{X}_1) \geq k_2$ *we say* $(\mathbf{X}_1, \mathbf{X}_2)$ *is a* (k_1, k_2)*-2-block source.*

By Lemma 2.6 one can prove that a high-entropy source is close to a block source.

Lemma 2.8 ([GW97])**.** *Let* $\mathbf{X}$ *be a* $(n_1 + n_2, n_1 + n_2 - \Delta)$*-source. Divide* $\mathbf{X}$ *into two blocks* $\mathbf{X} = \mathbf{X}_1 \circ \mathbf{X}_2$*, so that* $\mathbf{X}_1$ *has* n_1 *bits. Then* $(\mathbf{X}_1, \mathbf{X}_2)$ *is* ε*-close to a* $(k_1 - \Delta, k_2 - \Delta - \log(1/\varepsilon))$*-2-block source.*

Next we introduce an "average-case" variation of conditional min-entropy introduced by Dodis, Ostrovsky, Reyzin and Smith [DORS08] that is strong enough for many applications but have nicer properties than worst-case conditional min-entropy.

Definition 2.9 ([DORS08]). *For a joint distribution* $(\mathbf{X}, \mathbf{Z})$*, the* average conditional min-entropy *of* $\mathbf{X}$ *given* $\mathbf{Z}$ *is*

$$\widetilde{\mathrm{H}}_\infty(\mathbf{X} \mid \mathbf{Z}) := -\log\left(\mathop{\mathbb{E}}_{z\sim\mathbf{Z}}\left[\max_x(\Pr\left[\mathbf{X}=x \mid \mathbf{Z}=z\right])\right]\right).$$

The most important property is the following lemma, usually referred to as the *chain rule* (for min-entropy).

Lemma 2.10 ([DORS08]). *Let* $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$ *be (correlated) random variables. Then*

$$\widetilde{\mathrm{H}}_\infty(\mathbf{X} \mid (\mathbf{Y}, \mathbf{Z})) \geq \widetilde{\mathrm{H}}_\infty(\mathbf{X} \mid \mathbf{Z}) - \log(\mathrm{Supp}(\mathbf{Y})).$$

When we need to consider worst-case conditional min-entropy with only a guarantee on the average conditional min-entropy, we use the following lemma. Note that Lemma 2.10 and Lemma 2.11 directly imply Lemma 2.6.

Lemma 2.11 ([DORS08]). *Let* $\mathbf{X}, \mathbf{Z}$ *be (correlated) random variables. For every* $\varepsilon > 0$*,*

$$\Pr_{z\sim\mathbf{Z}}\left[\mathrm{H}_\infty(\mathbf{X}|_{\mathbf{Z}=z}) \geq \widetilde{\mathrm{H}}_\infty(\mathbf{X} \mid \mathbf{Z}) - \log(1/\varepsilon)\right] \geq 1-\varepsilon.$$

2.3 Seeded Extractors and Related Primitives

First we define (seeded) extractors and condensers, which are functions that take a source with some min-entropy and output a random variable that is close to a uniform or high-entropy string.

Definition 2.12. $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *is a* seeded extractor for entropy k with error ε *(or* (k,ε)-seeded extractor *for short) if for every* (n,k) *source* $\mathbf{X}$*, and every* $\mathbf{Y} = \mathbf{U}_d$*,*

$$\mathrm{Ext}(\mathbf{X}, \mathbf{Y}) \approx_\varepsilon \mathbf{U}_m.$$

We call d *the* seed length *of* Ext*. We say* Ext *is linear if* $\mathrm{Ext}(\cdot, y)$ *is a linear function for every* $y \in \{0,1\}^d$*. We say* Ext *is strong if*

$$(\mathrm{Ext}(\mathbf{X}, \mathbf{Y}) \approx_\varepsilon \mathbf{U}_m) \mid \mathbf{Y}.$$

Definition 2.13. *A function* $\mathrm{Con} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *is called a* (k, k', ε)-strong condenser *if for every independent* $\mathbf{X}, \mathbf{Y}$ *such that* $\mathbf{X}$ *is a* (n,k)*-source and* $\mathbf{Y} = \mathbf{U}_d$*, there exists* $\mathbf{Z}$ *such that*

$$(\mathrm{Con}(\mathbf{X}, \mathbf{Y}) \approx_\varepsilon \mathbf{Z}) \mid \mathbf{Y}$$

and $\mathrm{H}_\infty(\mathbf{Z} \mid \mathbf{Y}) \geq k'$*. In addition, we say* Con *is lossless if* $k = k'$*.*

We say Ext is linear if for every $s \in \{0,1\}^d$, $\mathrm{Ext}(\cdot, s)$ is a linear function over $\mathbb{F}_2$. Similarly we say Con is linear if for every $s \in \{0,1\}^d$, $\mathrm{Con}(\cdot, s)$ is a linear function over $\mathbb{F}_2$.

Next we introduce a few explicit extractors and condensers. The first one is the strong lossless condenser by Guruswami, Umans and Vadhan [GUV09] (the "GUV condenser") which has a linear instantiation as observed by Cheraghchi [Che10].

Lemma 2.14 ([GUV09],[Che10])**.** *For every* $n \in \mathbb{N}$*,* $k \leq n$*,* $\varepsilon > 0$ *and* $\alpha > 0$*, there is an explicit* (k, ε)*-strong linear lossless condenser* $\mathrm{Con} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *such that* $d = (1 + \frac{1}{\alpha}) \log(nk/\varepsilon) + O(1)$ *and* $m = (1+\alpha)k + d$*.*

We also need the strong seeded extractor based on the Leftover Hash Lemma, which can also be made linear.

Lemma 2.15 ([ILL89])**.** *For every* $n \in \mathbb{N}$*,* $k \leq n$*,* $\varepsilon > 0$*, there is an explicit* (k, ε)*-strong linear seeded extractor* $\mathrm{LHL} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *such that* $d = n$ *and* $m = k - 2\log(1/\varepsilon)$*.*

A simple composition of the above two primitives imply the following lemma.

Lemma 2.16 (condense-then-hash extractor)**.** *For every $n \in \mathbb{N}$, $k \leq n$, $\varepsilon > 0$, there is an explicit (k, ε)-strong linear seeded extractor $\mathrm{LExt} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ such that $k = m + 2\log(1/\varepsilon)$ and $d = 2m + 8\log(n/\varepsilon) + O(1)$.*

The following extractor is also by Guruswami, Umans and Vadhan [GUV09] (known as the "GUV extractor"). This extractor has parameters optimal up to a constant factor, but is not linear.

Lemma 2.17 ([GUV09])**.** *There exists a constant $\beta > 0$ such that for every $\varepsilon > 2^{-\beta n}$ and every k, there exists an explicit (k, ε)-strong seeded extractor $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ s.t. $d = O(\log(n/\varepsilon))$ and $m = k/2$.*

Next we define samplers.

Definition 2.18. *$\mathrm{Samp} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is an (ε, δ)-*(averaging) sampler *for entropy k if for every function $f : \{0,1\}^m \to [0,1]$ and every (n,k)-source $\mathbf{X}$,*

$$\Pr_{x \sim \mathbf{X}}\left[\left|\mathop{\mathbb{E}}_{s \in \{0,1\}^d}[f(\mathrm{Samp}(x,s))] - \mathop{\mathbb{E}}_{y \in \{0,1\}^m}[f(y)]\right| \leq \varepsilon\right] \geq 1 - \delta.$$

We say Samp is linear if $\mathrm{Samp}(\cdot, y)$ is linear for every $y \in \{0,1\}^d$. In addition, if we do not specify k then $k = n$ (i.e., $\mathbf{X}$ must be uniform) by default.

A celebrated result by Zuckerman [Zuc97] showed that one can use a seeded extractor as a sampler for weak sources.

Lemma 2.19 ([Zuc97])**.** *A $(k + \log(1/\delta), \varepsilon)$-seeded extractor is also an (ε, δ)-sampler for entropy k.*

Finally we define a disperser, which is a special case of samplers. In a disperser we also want to pick D samples from a set $[M]$, but we only want to make sure that not all D samples fall into an ε fraction of "bad" elements.

Definition 2.20. *We say a function* $\mathrm{Disp} : \{0,1\}^n \times [D] \to [M]$ *is a* (K, s)-disperser *if for every set* $T \subseteq \{0,1\}^m$ *of size* sM, $|\{r \in \{0,1\}^n : \forall j \in [D]\, \mathrm{Disp}(r, j) \in T\}| \leq K$.

For dispersers, Zuckerman [Zuc07] showed an explicit construction with very strong parameters.

Lemma 2.21 ([Zuc07])**.** *For every constant* $\gamma > 0$ *and every* $\varepsilon = \varepsilon(n) > 0$, *there exists an efficient family of* $(K = 2^{\gamma n}, s)$*-disperser* $\mathrm{Disp} : \{0,1\}^n \times [D] \to [M]$ *such that* $D = O(\frac{n}{\log(1/s)})$ *and* $M = \sqrt{K}$.

2.3.1 Standard Lemmas for Extractors

In this section we review some standard lemmas for extractors. The first one is the Leftover Hash Lemma (which implies the extractor in Lemma 2.15). The following form is more general than the original lemma in [ILL89], but is also standard in the literature. (See, e.g., [Vad12, Problem 6.3].)

Lemma 2.22 (Leftover Hash Lemma [ILL89])**.** *Consider any* $h : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *and any distribution* $\mathbf{Y} \in \{0,1\}^d$ *such that for every distinct* $x_1, x_2 \in \{0,1\}^n$, $\Pr_{y \sim \mathbf{Y}}[h(x_1, y) = h(x_2, y)] \leq (1+\varepsilon)2^{-m}$. *Then for any source* $\mathbf{X}$ *with min-entropy* $m + \log(1/\varepsilon)$,

$$(h(\mathbf{X}, \mathbf{Y})) \approx_{\sqrt{\varepsilon/2}} \mathbf{U}_m \mid \mathbf{Y}.$$

The second lemma is an improved analysis for applying seeded extractors on source that has conditional min-entropy. We need the slightly more general form as follows. We omit the proof of this general lemma because it is essentially the same as the original lemma in [Vad12, Problem 6.8].

Lemma 2.23. *Let $\mathbf{X} \leftrightarrow \mathbf{E} \leftrightarrow (\mathbf{Y}, \mathbf{Z})$ be a Markov chain such that $\mathbf{X} \in \mathcal{X}$ and $\mathbf{Y} \in \mathcal{S}$ are independent conditioned on $\mathbf{E}$, and $\widetilde{\mathrm{H}}_\infty(\mathbf{X} \mid \mathbf{E}) \geq k$. Let $\mathrm{Ext} : \mathcal{X} \times \mathcal{S} \to \{0,1\}^m$ be a function which satisfies the following conditions for an error parameter $\varepsilon > 0$ and a deterministic function g: for every $e \in \mathrm{Supp}(\mathbf{E})$ and every distribution $\mathbf{X}'$ of min-entropy k that is independent of $(\mathbf{Y}, \mathbf{Z})|_{\mathbf{E}=e}$ we have*

$$(\mathrm{Ext}(\mathbf{X}', \mathbf{Y}|_{\mathbf{E}=e}) \approx_\varepsilon \mathbf{U}_m) \mid (\mathbf{Z}|_{\mathbf{E}=e}).$$

Then

$$(\mathrm{Ext}(\mathbf{X}, \mathbf{Y}) \approx_{3\varepsilon} \mathbf{U}_m) \mid (\mathbf{E}, \mathbf{Z}).$$

The third lemma shows that block sources can be extracted with shorter seed length by composing two seeded extractors (given that $n_1 \gg n_2$ and $d_1 \gg d_2$).

Lemma 2.24 ([NZ96]). *Let $E_1 : \{0,1\}^{n_1} \times \{0,1\}^{d_1} \to \{0,1\}^m$ be a (k_1, ε_1) extractor and $E_2 : \{0,1\}^{n_2} \times \{0,1\}^{d_2} \to \{0,1\}^{d_1}$ be a (k_2, ε_2) extractor. Define $E((x_1, x_2), s) = E_1(x_1, E_2(x_2, s))$. Then for every (k_1, k_2)-block source $(X_1, X_2) \in \{0,1\}^{n_1} \times \{0,1\}^{n_2}$, $E((X_1, X_2), U_{d_2})$ is $(\varepsilon_1 + \varepsilon_2)$-close to $\mathbf{U}_m$.*

2.3.2 Space-efficient Sampler

In this section we construct a sampler (for full entropy) that is computable in linear space. This result is basically [Gol11, Theorem 6.1] with space complexity analysis.

Lemma 2.25. *For every $\delta, \varepsilon > 0$ and integer m, there exists a (ε, δ)-sampler $f : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ s.t. $d = O(\log\log(1/\delta) + \log(1/\varepsilon))$ and $n = m + O(\log(1/\delta)) + O(\log(1/\varepsilon))$. Moreover, for every x, y, $f(x, y)$ can be computed in space $O(m + \log(n/\varepsilon\delta))$.*

To prove this result, first we need a space-efficient seeded extractor for high-entropy sources constructed by Goldreich and Wigderson [GW97].

Lemma 2.26. *For every $\varepsilon, \Delta > 0$ and integer n there exists a $(n - \Delta, \varepsilon)$ extractor $E : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^n$ with $d = O(\Delta + \log(1/\varepsilon))$, and for every x, y, $E(x, y)$ can be computed in space $O(\log(n/\varepsilon))$.*

We also need a space complexity bound for the GUV extractor (Lemma 2.17) by Kane, Nelson and Woodruff [KNW08].

Claim 2.27. *The extractor from Lemma 2.17 is computable in $O(n)$ space.*

Now we are ready to prove Lemma 2.25.

Proof of Lemma 2.25. Let $\Delta = \log(1/\delta) + 1$. Let $E_1 : \{0,1\}^m \times \{0,1\}^{d_1} \to \{0,1\}^m$ be an $(m - \Delta, \varepsilon/3)$-seeded extractor from Lemma 2.26. Then let $E_2 : \{0,1\}^{3d_1} \times \{0,1\}^d \to \{0,1\}^{d_1}$ be an $(2d_1, \varepsilon/3)$-seeded extractor from Lemma Lemma 2.17. Then we claim that $E : \{0,1\}^{m+3d_1} \times \{0,1\}^d \to \{0,1\}^m$, defined as $E((x_1, x_2), s) = E_1(x_1, E_2(x_2, s))$, is a $(m + 3d_1 - \Delta, \varepsilon)$-seeded extractor, and thus a (ε, δ) sampler by Lemma 2.19.

To prove the claim, consider any $(m + 3d_1 - \Delta)$-source $\mathbf{X}$. By Lemma 2.8, $\mathbf{X}$ is $(\varepsilon/3)$-close to a $(m-\Delta, 3d_1-\Delta-\log(3/\varepsilon))$-block source $(\mathbf{X}_1, \mathbf{X}_2) \in \{0,1\}^m \times \{0,1\}^{3d_1}$. By Lemma 2.24, $E_1(\mathbf{X}_1, E_2(\mathbf{X}_2, \mathbf{U}_d))$ is $2\varepsilon/3$-close to uniform. Since $E(\mathbf{X}, \mathbf{U}_d)$ is $\varepsilon/3$-close to $E_1(\mathbf{X}_1, E_2(\mathbf{X}_2, \mathbf{U}_d))$, by triangle inequality it is ε-close to uniform. Moreover, it suffices to take $d = O(\log(d_1/\varepsilon)) = O(\log\log(1/\delta) + \log(1/\varepsilon))$, $n = m + 3d_1 = m + O(\log(1/\delta) + \log(1/\varepsilon))$, and the space complexity of E is $O(n) = O(m + \log(1/\varepsilon) + \log(1/\delta))$. □

2.4 Markov Chain and Noisy Extraction

In this thesis we usually consider the scenario that we have two sources $\mathbf{X}, \mathbf{Y}$ which are independent conditioned on a random variable $\mathbf{Z}$. We use Markov chain as a shorthand for this.

Definition 2.28. *Let $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$ be random variables. We say $\mathbf{X} \leftrightarrow \mathbf{Z} \leftrightarrow \mathbf{Y}$ forms a Markov chain (or $\mathbf{X} \leftrightarrow \mathbf{Z} \leftrightarrow \mathbf{Y}$ for short) if $\mathbf{X}$ and $\mathbf{Y}$ are independent conditioned on any fixing of $\mathbf{Z}$.*

Note that $\mathbf{X}$ and $\mathbf{Y}$ are symmetric in the above definition. Therefore all the properties stated below should also hold for its symmetric variant. Besides, all the properties stated below hold for the degenerate case that $\mathbf{X}$ is empty.

Lemma 2.29. *If $\mathbf{X} \leftrightarrow \mathbf{Z} \leftrightarrow \mathbf{Y}$ is a Markov chain, then for every deterministic function f, let $\mathbf{W} = f(\mathbf{X}, \mathbf{Z})$. Then*

- *$(\mathbf{X}, \mathbf{W}) \leftrightarrow \mathbf{Z} \leftrightarrow \mathbf{Y}$ is a Markov chain.*
- *$\mathbf{X} \leftrightarrow (\mathbf{W}, \mathbf{Z}) \leftrightarrow \mathbf{Y}$ is a Markov chain.*

Sometimes we use "$\mathbf{W}$ is a deterministic function of $\mathbf{X}$ (conditioned on $\mathbf{Z}$)" to refer to the first item, and "fix $\mathbf{W}$" to refer to the second item.

In addition, we usually face the case that we need to treat a random variable that is only close to uniform as an actual uniform random variable, and at the same time we need to maintain the same independence in a Markov Chain. To analyze error in this case, it will be convenient to use the classic coupling lemma:

Lemma 2.30 (coupling lemma [Ald83])**.**

1. *For every marginal distributions* $\mathbf{X}, \mathbf{Y}$ *over the same sample space, there exists a joint distribution* $(\mathbf{X}, \mathbf{Y})$ *such that* $\Pr[\mathbf{X} \neq \mathbf{Y}] = 2\Delta(\mathbf{X}; \mathbf{Y})$
2. *Every two (correlated) random variables* $\mathbf{X}, \mathbf{Y}$ *over the same sample space satisfy* $2\Delta(\mathbf{X}; \mathbf{Y}) \leq \Pr[\mathbf{X} \neq \mathbf{Y}]$.

Corollary 2.31. *For every pair of joint distributions* $(\mathbf{X}, \mathbf{Z}), (\mathbf{Y}, \mathbf{Z})$ *there exists a joint distribution* $(\mathbf{X}, \mathbf{Y}, \mathbf{Z})$ *such that* $\Pr[\mathbf{X} \neq \mathbf{Y}] = 2\Delta((\mathbf{X}, \mathbf{Z}); (\mathbf{Y}, \mathbf{Z}))$.

Proof. For any z we couple $\mathbf{X}|_{\mathbf{Z}=z}$ with $\mathbf{Y}|_{\mathbf{Z}=z}$ such that $\Pr[\mathbf{X} \neq \mathbf{Y} \mid \mathbf{Z} = z] = 2\Delta(\mathbf{X}|_{\mathbf{Z}=z}; \mathbf{Y}|_{\mathbf{Z}=z})$. Then

$$\Pr[\mathbf{X} \neq \mathbf{Y}] = \mathop{\mathbb{E}}_{z \sim \mathbf{Z}}[\mathbf{X} \neq \mathbf{Y} \mid \mathbf{Z} = z] = 2 \mathop{\mathbb{E}}_{z \sim \mathbf{Z}}[\Delta(\mathbf{X}|_{\mathbf{Z}=z}; \mathbf{Y}|_{\mathbf{Z}=z})] = 2\Delta((\mathbf{X}, \mathbf{Z}); (\mathbf{Y}, \mathbf{Z})).$$

□

A simple application of the coupling lemma would is the following lemma which shows that we can lift "local closeness" to "global closeness". (A similar lemma can be found in [Li15, Lemma 3.20].)

Lemma 2.32. *Let* $(\mathbf{X}, \mathbf{W}) \leftrightarrow \mathbf{Z} \leftrightarrow \mathbf{Y}$ *be a Markov chain. Suppose there exists a joint distribution* $(\widetilde{\mathbf{W}}, \mathbf{Z})$ *such that* $(\widetilde{\mathbf{W}} \approx_\delta \mathbf{W}) \mid \mathbf{Z}$. *Then there exists a joint distribution* $(\mathbf{X}, \widetilde{\mathbf{W}}, \mathbf{Z}, \mathbf{Y})$ *such that* $(\widetilde{\mathbf{W}} \approx_\delta \mathbf{W}) \mid (\mathbf{Z}, \mathbf{X}, \mathbf{Y})$ *and* $(\mathbf{X}, \widetilde{\mathbf{W}}) \leftrightarrow \mathbf{Z} \leftrightarrow \mathbf{Y}$.

Proof. We define the joint distribution $(\mathbf{X}, \mathbf{W}, \widetilde{\mathbf{W}}, \mathbf{Z}, \mathbf{Y})$ as follows. By the corollary above, there is a joint distribution such that $(\widetilde{\mathbf{W}}, \mathbf{W}, \mathbf{Z})$ such that $\Pr\left[\widetilde{\mathbf{W}} \neq \mathbf{w}\right] = 2\Delta\left((\widetilde{\mathbf{W}}, \mathbf{Z}); (\mathbf{W}, \mathbf{Z})\right) \leq 2\delta$. Conditioned on $(\mathbf{W}, \mathbf{Z})$, we sample $\widetilde{\mathbf{W}}$ based on this joint distribution and independent of any other random variables. Because $(\mathbf{X}, \mathbf{W}) \leftrightarrow \mathbf{Z} \leftrightarrow \mathbf{Y}$, and $\widetilde{\mathbf{W}}$ only depends on $(\mathbf{W}, \mathbf{Z})$, we also have $(\mathbf{X}, \widetilde{\mathbf{W}}) \leftrightarrow \mathbf{Z} \leftrightarrow \mathbf{Y}$. Furthermore, because $\Pr\left[(\mathbf{W}, \mathbf{Z}, \mathbf{X}, \mathbf{Y}) \neq (\widetilde{\mathbf{W}}, \mathbf{Z}, \mathbf{X}, \mathbf{Y})\right] = \Pr\left[\mathbf{W} \neq \widetilde{\mathbf{W}}\right] \leq 2\delta$, by the second part of coupling lemma we can conclude that $(\widetilde{\mathbf{W}} \approx_\delta \mathbf{W}) \mid (\mathbf{Z}, \mathbf{X}, \mathbf{Y})$. □

Another application is the following lemma which considers functions that are in some sense similar to strong seeded extractors. Note that in a trivial proof we need to switch $\mathbf{W}$ to $\widetilde{\mathbf{W}}$ in the input, and then switch $\widetilde{\mathbf{W}}$ back to $\mathbf{W}$ in the output, which incurs a 2δ error in total. Such a blow-up is not ideal if we need to apply the same argument for many times. The coupling lemma gives a simple proof to the following tighter bound.

Lemma 2.33. *Suppose $\mathbf{W} \approx_\delta \widetilde{\mathbf{W}}$. Let f, g be deterministic functions such that for some $\mathbf{X}$ independent of $\widetilde{\mathbf{W}}$ it holds that $f(\mathbf{X}, \widetilde{\mathbf{W}}) \approx_\varepsilon \mathbf{U} \mid (\widetilde{\mathbf{W}}, g(\mathbf{X}, \widetilde{\mathbf{W}}))$. Then for independent $\mathbf{X}$ and $\mathbf{W}$, $f(\mathbf{X}, \mathbf{W}) \approx_{\varepsilon+\delta} \mathbf{U} \mid (\mathbf{W}, g(\mathbf{X}, \mathbf{W}))$*

Proof. By the first part of coupling lemma we can find a joint distribution $(\mathbf{X}, \mathbf{W}, \widetilde{\mathbf{W}})$ such that $\Pr\left[\mathbf{W} \neq \widetilde{\mathbf{W}}\right] \leq 2\delta$, and $(\mathbf{W}, \widetilde{\mathbf{W}})$ is independent of $\mathbf{X}$. Observe that after fixing $\widetilde{\mathbf{W}}$, $(f(\mathbf{X}, \widetilde{\mathbf{W}}), g(\mathbf{X}, \widetilde{\mathbf{W}}))$ become deterministic functions of $\mathbf{X}$, so $(\mathbf{X}, f(\mathbf{X}, \widetilde{\mathbf{W}}), g(\mathbf{X}, \widetilde{\mathbf{W}})) \leftrightarrow \widetilde{\mathbf{W}} \leftrightarrow \mathbf{W}$ is a Markov chain. By the first part of coupling lemma, there is a joint distribution $(\widetilde{\mathbf{Y}}, f(\mathbf{X}, \widetilde{\mathbf{W}}), g(\mathbf{X}, \widetilde{\mathbf{W}}), \widetilde{\mathbf{W}})$ such that $\widetilde{\mathbf{Y}}$ is uniform conditioned on $(g(\mathbf{X}, \widetilde{\mathbf{W}}), \widetilde{\mathbf{W}})$ and $\Pr\left[f(\mathbf{X}, \widetilde{\mathbf{W}}) \neq \mathbf{Y}\right] \leq 2\varepsilon$. We sample $\widetilde{\mathbf{Y}}$ based on only $(f(\mathbf{X}, \widetilde{\mathbf{W}}), g(\mathbf{X}, \widetilde{\mathbf{W}}), \widetilde{\mathbf{W}})$ so that $(\mathbf{X}, f(\mathbf{X}, \widetilde{\mathbf{W}}), g(\mathbf{X}, \widetilde{\mathbf{W}}), \widetilde{\mathbf{Y}}) \leftrightarrow \widetilde{\mathbf{W}} \leftrightarrow \mathbf{W}$. Therefore $\widetilde{\mathbf{Y}}$ is uniform even when conditioned on $(g(\mathbf{X}, \widetilde{\mathbf{W}}), \widetilde{\mathbf{W}}, \mathbf{W})$. Now we define $\mathbf{Y}$ to be the random variable such that $\mathbf{Y} = \widetilde{\mathbf{Y}}$ if $\widetilde{\mathbf{W}} = \mathbf{W}$, or $\mathbf{Y}$ is an independent uniform random variable otherwise. Observe that for any fixing of $(\widetilde{\mathbf{W}}, \mathbf{W})$ such that $\widetilde{\mathbf{W}} = \mathbf{W}$, $\mathbf{Y}$ is uniform when conditioned on any fixing of $g(\mathbf{X}, \mathbf{W})$ because $\widetilde{\mathbf{Y}}$ is uniform conditioned on any fixing of $g(\mathbf{X}, \widetilde{\mathbf{W}})$, and $\widetilde{\mathbf{W}} = \mathbf{W}$. In addition, for any fixing of $(\widetilde{\mathbf{W}}, \mathbf{W})$ such that $\widetilde{\mathbf{W}} \neq \mathbf{W}$, $\mathbf{Y}$ is clearly uniform conditioned on $g(\mathbf{X}, \mathbf{W})$. Therefore $(\mathbf{Y} = \mathbf{U}) \mid (\mathbf{W}, g(\mathbf{X}, \mathbf{W}))$. Finally note that $\Pr[f(\mathbf{X}, \mathbf{W}) \neq \mathbf{Y}] \leq \Pr\left[\mathbf{W} \neq \widetilde{\mathbf{W}} \vee f(\mathbf{X}, \widetilde{\mathbf{W}}) \neq \widetilde{\mathbf{Y}}\right] \leq 2\delta + 2\varepsilon$. The claim follows by the second part of coupling lemma. □

One can generalize the lemma above to Markov chains $\mathbf{X} \leftrightarrow \mathbf{Z} \leftrightarrow \mathbf{W}$, $\mathbf{X} \leftrightarrow \mathbf{Z} \leftrightarrow \widetilde{\mathbf{W}}$ and $(\mathbf{Z}, \mathbf{W}) \approx_\delta (\mathbf{Z}, \widetilde{\mathbf{W}})$ by simply considering all fixings of $\mathbf{Z}$ and taking the average. Throughout this thesis, the most common special case we consider is the following lemma from [CS16a, Lemma 3.11] (which further includes an application of Lemma 2.11).

Lemma 2.34. *Let $\mathbf{X} \leftrightarrow \mathbf{Z} \leftrightarrow (\mathbf{Y}, \mathbf{W})$ be a Markov chain such that $(\mathbf{W} \approx_\delta \mathbf{U}_d) \mid \mathbf{Z}$ and $\widetilde{\mathrm{H}}_\infty(\mathbf{X} \mid \mathbf{Z}) \geq k + \log(1/\varepsilon)$. Let $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ be a (k, ε)-strong seeded extractor. Then*

$$(\mathrm{Ext}(\mathbf{X}, \mathbf{W}) \approx_{2\varepsilon+\delta} \mathbf{U}_m) \mid (\mathbf{Z}, \mathbf{Y}, \mathbf{W}).$$

2.5 Fourier Analysis

First we recall some basic definitions and properties in Fourier analysis.

Definition 2.35. *Let $f : \mathbb{F}_2^n \to \mathbb{R}$ be a function. The* Fourier coefficients *of f, denoted by $\widehat{f} : \mathbb{F}_2^n \to \mathbb{R}$, are*

$$\widehat{f}(\alpha) := \mathop{\mathbb{E}}_{x \sim \mathbb{F}_2^n} \left[f(x) \cdot (-1)^{\langle \alpha, x \rangle} \right].$$

Lemma 2.36 (Parseval-Plancherel identity)**.** *For every functions $f, g : \mathbb{F}_2^n \to \mathbb{R}$,*

$$\mathop{\mathbb{E}}_{x \sim \mathbb{F}_2^n} [f(x) g(x)] = \sum_{\alpha \in \mathbb{F}_2^n} \widehat{f}(\alpha) \widehat{g}(\alpha).$$

Definition 2.37. *The convolution of functions $f, g : \mathbb{F}_2^n \to \mathbb{R}$, denoted by $f * g : \mathbb{F}_2^n \to \mathbb{R}$, is defined as*

$$f * g(x) := \mathop{\mathbb{E}}_{y \sim \mathbb{F}_2^n} [f(y) g(x - y)].$$

Lemma 2.38. *For every functions $f, g : \mathbb{F}_2^n \to \mathbb{R}$ and every $\alpha \in \mathbb{F}_2^n$,*

$$\widehat{f * g}(\alpha) = \widehat{f}(\alpha) \widehat{g}(\alpha).$$

Next we define a density function.

Definition 2.39. *For every $A \subseteq \mathbb{F}_2^n$, define the density function of A to be $\mu_A := \frac{2^n}{|A|} \cdot \mathbb{1}_A$. For a distribution $\mathbf{A}$ on $\mathbb{F}_2^n$, the density function of $\mathbf{A}$, denoted by $\mu_{\mathbf{A}}$, is defined as $\mu_{\mathbf{A}}(x) = 2^n \Pr[\mathbf{A} = x]$.*

We need the following three facts about density functions.

Lemma 2.40. *Let $f : \mathbb{F}_2^n \to \mathbb{R}$ be a function and let $\mathbf{A}$ be a distribution on $\mathbb{F}_2^n$. Then*

$$\mathbb{E}[f(\mathbf{A})] = \mathop{\mathbb{E}}_{x \sim \mathbb{F}_2^n}[\mu_{\mathbf{A}}(x) f(x)].$$

Lemma 2.41. *Let $\mathbf{A}, \mathbf{B}$ be two distributions on $\mathbb{F}_2^n$. Then $\mu_{\mathbf{A}+\mathbf{B}} = \mu_{\mathbf{A}} * \mu_{\mathbf{B}}$.*

Lemma 2.42. *If $V \subseteq \mathbb{F}_2^n$ is a linear subspace, then $\widehat{\mu_V}(\alpha) = 1$ if $\alpha \in V^\perp$ and $\widehat{\mu_V}(\alpha) = 0$ otherwise.*

Finally we need Chang's lemma.

Lemma 2.43 ([Cha02]). *For $X \subseteq \mathbb{F}_2^n$, define $\mathrm{Spec}_\gamma(X) = \{\alpha \in \mathbb{F}_2^n : |\widehat{\mu_X}(\alpha)| \geq \gamma\}$. Define $\beta = |X| / |\mathbb{F}_2^n|$. Then*

$$\dim(\mathrm{span}(\mathrm{Spec}_\gamma(X))) \leq 2\gamma^{-2} \ln(1/\beta).$$

2.6 Matrices

Throughout this thesis, for a matrix $M \in \mathbb{R}^{w \times w}$ we use $M[i, j]$ to denote the entry in the i-th row and j-th column of M. For a column vector $v \in \mathbb{R}^w$ we use $v[i]$ to denote its i-th element. Next we define a few matrix norms that we will use.

Definition 2.44. *Let M be a $w \times w$ real matrix. The matrix norms of M are defined as follows.*

- *Max norm:* $\|M\|_{\max} := \max_{i,j} M[i,j]$
- *Infinity norm:* $\|M\|_\infty := \max_i(\sum_j M[i,j])$.
  Equivalently, $\|M\|_\infty : \sup_{x\neq 0}(\|Mx\|_\infty / \|x\|_\infty)$
- *2-norm (or norm for short):* $\|M\| := \sup_{x\neq 0}(\|Mx\| / \|x\|)$.

Definition 2.45. *We say a matrix M is (right)* stochastic *if* $\min_{i,j} M[i,j] \geq 0$ *and* $\forall i \sum_j M[i,j] = 1$. *We say M is doubly stochastic if it further satisfies that* $\forall j \sum_i M[i,j] = 1$.

The following are some basic properties of matrix norm.

Fact 2.46. *Let M_1, M_2 be two $w \times w$ matrices. Then*

- *Sub-additivity:* $\|M_1 + M_2\| \leq \|M_1\| + \|M_2\|$
- *Sub-multiplicativity:* $\|M_1 M_2\| \leq \|M_1\| \, \|M_2\|$
- *Sub-additivity:* $\|M_1 + M_2\|_\infty \leq \|M_1\|_\infty + \|M_2\|_\infty$
- *Sub-multiplicativity:* $\|M_1 M_2\|_\infty \leq \|M_1\|_\infty + \|M_2\|_\infty$
- *If M_1 is stochastic, then* $\|M_1\|_\infty = 1$.
- *If M_1 is doubly stochastic, then* $\|M_1\| \leq 1$.

Lemma 2.47. *Let M_1, M_2, M_1', M_2' be $w \times w$ matrices with infinity norm at most* 1.

$$\|M_1' M_2' - M_1 M_2\|_\infty \leq \|M_1' - M_1\|_\infty + \|M_2' - M_2\|_\infty .$$

Proof.

$$\begin{aligned}
\|M_1' M_2' - M_1 M_2\|_\infty &\leq \|M_1' M_2 - M_1 M_2\|_\infty + \|M_1' M_2' - M_1' M_2\|_\infty \\
&\leq \|M_1' - M_1\|_\infty \, \|M_2\|_\infty + \|M_1'\|_\infty \, \|M_2' - M_2\|_\infty \\
&\leq \|M_1' - M_1\|_\infty + \|M_2' - M_2\|_\infty .
\end{aligned}$$

□

2.7 Space Complexity

In this section we review some basic facts about the analysis of space complexity. We consider the standard model which is a Turing machine with a read-only input tape, a read-write work tape, and a write-only output tape. We say an algorithm runs in space $S(n)$ if the *work tape* head touches at most $S(n)$ cells throughout the computation, where n denotes the input length. Throughout this thesis we only consider algorithm that runs in space at least $S(n) \geq \log n$.

Note that the input and output length of an algorithm can be much larger than its space complexity $S(n)$. In this case the composition of two algorithms F_1, F_2 should not be done in the trivial way (i.e., on input x, compute $F_2(x)$ in the work tape and use it as the input of F_1). Nevertheless, one can still apply the standard trick of computing $F_2(x)$ *on the fly*. That is, suppose the space complexity of F_1 and F_2 are whenever the simulation of F_1 attempts to read the i-th symbol of $F_2(x)$, denoted by $F_2(x)_i$, one can pause the simulation and compute the mapping $(x, i) \mapsto F_2(x)_i$, which takes only $S_2(n)+O(\log(m))$ extra space, where $n = |x|, m = |F_2(x)|$ and $S_2(n)$ is the space-complexity of F_2. (See, e.g., [AB09, Lemma 4.15].) Formally, we have the following lemma for the space complexity of the composition of two functions.

Lemma 2.48. *Let F_1, F_2 be functions that have space complexity $S_1(n), S_2(n) \geq \log(n)$ on input length n respectively, and F_2 has output length $m(n)$. Then the composition of functions $F_1(F_2(\cdot))$ has space complexity $O(S_1(m(n)) + S_2(n))$.*

The same trick can be generalized to iterated composition of multiple functions, and we will need a tighter analysis as follows. For convenience, first we assume without loss of generality that every Turing machine must "read proactively" in the following sense. Each Turing machine has a specific READ operation that writes

the symbol touched by the input tape head into the register. Furthermore, for any operation that is not READ, the behavior of the Turing machine is independent of the input tape content. With this assumption, it makes sense to discuss the maximum number of cells that is occupied whenever F reads from the input head, which we call the *sensitive space complexity*. If F has sensitive space complexity $S^*(n)$ that is much smaller than its space complexity $S(n)$, then the t-iterated composition of F has space complexity $O(t \cdot S^*(n) + S(n))$ instead of $O(t \cdot S(n))$, because we only need to keep $S^*(n)$ cells protected when we recursively compute the input on the fly. This fact also generalizes to the case that the function F takes an auxiliary input in each iteration of composition. This idea can be formalized with the following lemma from Hoza's dissertation [Hoz21b, Lemma 2.1.1].

Lemma 2.49 ([Hoz21b]). *Let $F : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ be a function that satisfies $|F(x,y)| = |x|$ for every x, y. Suppose for any $x \in \{0,1\}^n, y \in \{0,1\}^r$, F has space complexity $S(n,r)$. Furthermore, suppose that during the computation of $F(x,y)$, whenever F reads a bit from x, only the first $S^*(n,r)$ cells on the work tape are occupied. Now define the t-iterated composition of F, denoted by $F^{(t)}$, with the following recursive formula:*

$$F^{(0)}(x) = x,$$
$$F^{(i)}(x, y_1, \ldots, y_i) = F(F^{(i-1)}(x, y_1, \ldots, y_{i-1}), y_i).$$

Then $F^{(t)}$ has space complexity $O(S(n,r) + t \cdot S^*(n,r))$.

Finally, we also need the following fact that arithmetic operations can be done in logarithmic space.

Lemma 2.50 ([BCH86, CDL01]). *Let $x_1, x_2, \ldots, x_n$ be n-bit integers. Then $-x_1, \sum_i x_i, \prod_i x_i, \lfloor x_1/x_2 \rfloor$ can all be computed in $O(\log(n))$ space.*

Combine the lemma above with an idea similar to Savitch's [Sav70] theorem, one

can also bound the space complexity of iterated matrix multiplications.

Lemma 2.51 ([MRSV17, AKM+20]). *Let $\mathbf{M}_1, \ldots, \mathbf{M}_t$ be $w \times w$ real matrices where each entry has bit length at most T. Then $\prod_{i=1}^{t} \mathbf{M}_i$ can be computed in space $O(\log(t)\log(twT))$.*

Chapter 3

Chattopadhyay-Zuckerman Framework and Correlation Breakers

In this chapter, we give a recap of the two-step framework for constructing two-source extractors by Chattopadhyay and Zuckerman [CZ19]. Our construction of affine extractors (Chapter 4) and sumset extractors (Chapter 5) will also be based on this framework. Then we will introduce the notion of correlation breakers [Li13, Coh16a], which also plays a crucial role in our extractor constructions, and review a few related ideas in the literature.

This chapter is an exposition of ideas from prior works, and the reformulation of some prior ideas that we show in this chapter first appeared in the following joint work:

- [CGL21] Eshan Chattopadhyay, Jesse Goodman, and Jyun-Jie Liao. Affine extractors for almost logarithmic entropy. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021*, pages 622–633. IEEE, 2021

3.1 NOBF Sources

The Chattopadhyay-Zuckerman framework (or CZ framework for short) consists of two steps: first convert the given source to a "non-oblivious bit-fixing source" (NOBF source), then apply an extractor for NOBF sources. In this section we introduce the definition of NOBF sources and two explicit extractors for NOBF sources.

Definition 3.1. *A distribution $\mathbf{X} = (\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_n)$ on $\{0,1\}^n$ is called a (q,t)-NOBF source if there exists a set Q s.t. $|Q| \leq q$ and $\mathbf{X}_{[n]\setminus Q}$ is t-wise independent.*

In other words, a (q,t)-NOBF source is a distribution over n-bit strings that will become t-wise independent if we remove q "bad bits" from the string. Note that the indices of the bad bits are unknown, and the name "non-oblivious" comes from the fact that the bad bits can be arbitrarily correlated with the good bits (as opposed to "oblivious" bit-fixing sources [CGH+85] where the bad bits can be treated as constants).

There are two known explicit constructions of extractors for NOBF sources. The first one is simply the majority function, which works for $t = O(1)$, but only has constant error and requires $q \ll \sqrt{n}$.

Lemma 3.2 ([DGJ+10, Vio14])**.** *For every $\varepsilon > 0$, the majority function $\mathrm{Maj} : \{0,1\}^n \to \{0,1\}$ is an extractor for (q,t)-NOBF sources with error $\varepsilon + O(n^{-0.1})$ where $q = n^{0.4}$ and $t = O(\varepsilon^{-2}\log^2(1/\varepsilon))$.*

The second one is constructed by Chattopadhyay and Zuckerman [CZ19], which can work for any $q = n^{1-\Omega(1)}$ and polynomially small error, but requires the independence t to be $\mathrm{polylog}(n)$. (This cause the corresponding two-source extractors to require $\mathrm{polylog}(n)$ entropy.) Their construction was later improved by Li [Li16]

to get longer output, and by Meka [Mek17] to get smaller constant exponent in the independence number t.

Lemma 3.3 ([CZ19, Li16, Mek17]). *There exists an explicit function* $\mathrm{BFExt} : \{0,1\}^n \to \{0,1\}^m$ *for* (q,t)*-NOBF sources with error* $n^{-\Omega(1)}$ *where* $m = n^{\Omega(1)}$, $q = n^{0.9}$ *and* $t = (m\log(n))^{C_{3.3}}$ *for some constant* $C_{3.3}$.

3.2 Correlation Breakers

Next we discuss more about the first step: converting the given source to an NOBF source. We will review a few ideas that were used to convert two independent sources into an NOBF source in the literature, and these ideas will also be useful in our construction of affine extractors and sumset extractors. In a high-level view, one can turn two independent sources $(\mathbf{X},\mathbf{Y}) \in \{0,1\}^n \times \{0,1\}^n$ into an NOBF source in two steps. First of all, take a strong seeded extractor $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ with error $\varepsilon/2$ and enumerate all $D = 2^d$ seeds $i \in [D]$ and compute $\mathbf{Y}_i := \mathrm{Ext}(\mathbf{Y}, i)$. One can use Markov's inequality to show that $\mathbf{Y}_i$ is close to uniform for most of i, and for simplicity we assume that $\mathbf{Y}_i$ is exactly uniform for $1-\varepsilon$ fraction of i.[5] Now we get a sequence of random variables $(\mathbf{Y}_1, \mathbf{Y}_2, \ldots, \mathbf{Y}_D)$ such that $\mathbf{Y}_i$ is uniform for all except ε fraction of $i \in [D]$, but we don't know exactly which $\mathbf{Y}_i$ is uniform, and these random variables can be arbitrarily correlated. Intuitively one can view $(\mathbf{Y}_1, \ldots, \mathbf{Y}_D)$ as a $(\varepsilon D, 1)$-NOBF source. The next step is to use the other independent source $\mathbf{X}$ to "break the correlations" between different $\mathbf{Y}_i$s and increase the independence to t. This step can be done with a primitive called *correlation breakers* (with advice), which first appeared in a multi-source extractor construction

[5] This is of course not true, but it helps provide better intuition about the framework. In Chapter 5 we will revisit some issues that we will face after removing this assumption.

by Li [Li13]. Briefly speaking, this primitive takes the sequence $(\mathbf{Y}_1, \mathbf{Y}_2, \ldots, \mathbf{Y}_D)$ and an independent source $\mathbf{X}$, and output a sequence $(\mathbf{Z}_1, \ldots, \mathbf{Z}_D)$ so that any t of the $\mathbf{Z}_i$s becomes independent of each other. Formally, a correlation breaker is defined as follows.

Definition 3.4. $\mathrm{CB} : \{0,1\}^n \times \{0,1\}^d \times \{0,1\}^a \to \{0,1\}^m$ *is a t-correlation breaker for entropy k with error ε (or a (t, k, ε)-correlation breaker for short) if for every distributions $\mathbf{X} \in \{0,1\}^n$, $\mathbf{S}, \mathbf{S}_1, \ldots, \mathbf{S}^t \in \{0,1\}^d$ and strings $\alpha, \alpha^1, \ldots, \alpha^t \in \{0,1\}^a$ such that*

- $\mathrm{H}_\infty(\mathbf{X}) \geq k$ *and* $\mathbf{S}$ *is uniform*
- $\mathbf{X}$ *is independent of* $(\mathbf{S}, \mathbf{S}^{[t]})$
- $\forall i \in [t]$, $\alpha \neq \alpha^i$,

it holds that

$$\mathrm{CB}(\mathbf{X}, \mathbf{S}, \alpha) \approx_\varepsilon \mathbf{U}_m \mid \{\mathrm{CB}(\mathbf{X}, \mathbf{S}^i, \alpha^i)\}_{i \in [t]}.$$

We call $\mathbf{S}$ *the seed of* CB*, and we say* CB *is strong if the same condition holds even after conditioning on the seeds, i.e.,*

$$\mathrm{CB}(\mathbf{X}, \mathbf{S}, \alpha) \approx_\varepsilon \mathbf{U}_m \mid \left(\mathbf{S}, \{\mathbf{S}^i, \mathrm{CB}(\mathbf{X}, \mathbf{S}^i, \alpha^i)\}_{i \in [t]}\right).$$

We briefly explain why a correlation breaker in the above form can achieve the task as claimed. Assume for simplicity that we have a correlation breaker CB that has 1-bit output and 0 error (which of course does not actually exist). Then we can compute $\mathbf{Z}_i := \mathrm{CB}(\mathbf{X}, \mathbf{Y}_i, i)$ for every $i \in \{0,1\}^d$, and get the following guarantee for the D-bit string $(\mathbf{Z}_1, \ldots, \mathbf{Z}_D)$: if $\mathbf{Y}_i$ is uniform, then $\mathbf{Z}_i$ is also uniform even when conditioned on any t other bits. Therefore, if we remove the εD "bad" $\mathbf{Y}_i$s that are not uniform and the corresponding "bad" $\mathbf{Z}_i$s, we get a $(t+1)$-wise independent distribution. This means $(\mathbf{Z}_1, \ldots, \mathbf{Z}_D)$ is a $(\varepsilon D, t+1)$-NOBF source.

Note that in the definition of correlation breakers, we want to show that the function $\mathrm{CB}(\mathbf{X}, \cdot, \alpha)$ applied on a uniform $\mathbf{S}$ would output a random variable that is uniform even when conditioned on the outputs of the same function (but with different advice) on other correlated random variables $\mathbf{S}^1, \ldots, \mathbf{S}^t$. A similar task first appeared in the setting of *non-malleable extractors* introduced by Dodis and Wichs [DW09],[6] and in the context of non-malleable extractors $\mathbf{S}^1, \ldots, \mathbf{S}^t$ are considered to be "tampered versions" of $\mathbf{S}$ that are generated by some adversary. While the motivation is different here, we will still call $\mathbf{S}^i$ a tampered version of $\mathbf{S}$, and correspondingly call $f(\mathbf{S}^i, \alpha^i)$ a tampered version of $f(\mathbf{S}, \alpha)$ for any function f. In addition, when we define some new random variable $\mathbf{R} := f(\mathbf{S}, \alpha)$, we always assume that its tampered version $f(\mathbf{S}^i, \alpha^i)$ is denoted by $\mathbf{R}^i$.

3.3 Alternating Extraction

Next we review the concept of alternating extraction [DP07], which was first used in [DW09] as a way to break correlation from the tampered versions. The basic idea is simple: if $\mathbf{R}$ is the output of a strong seeded extractor $\mathrm{Ext}(\mathbf{X}, \mathbf{Y})$, then $\mathbf{R}$ is uniform even when conditioned on $\mathbf{Y}$. After conditioning on $\mathbf{Y}$, $\mathbf{R}$ becomes a deterministic function of $\mathbf{X}$ and thus is independent of $\mathbf{Y}^{[t]}$, the tampered versions of $\mathbf{Y}$. Therefore $\mathbf{R}$ is uniform conditioned on $\mathbf{Y}^{[t]}$. Note that this does not give us a correlation breaker yet, because $\mathbf{Y}^{[t]}$ are tampered versions of $\mathbf{Y}$, but not tampered versions of $\mathbf{R}$. However, we can use the additional advice string to help us solve this problem. Before we discuss how advice strings can help, first we introduce the alternating extraction process, which extends the above idea for multiple rounds

[6] In fact, a reduction in [CGL20] shows that correlation breakers and non-malleable extractors are equivalent.

to generate a sequence of random variables $\mathbf{S}_0, \mathbf{R}_1, \mathbf{S}_1, \mathbf{R}_2, \mathbf{S}_2, \ldots$ such that every random variables is uniform even when conditioned on previous random variables *and their tampered versions*.

Lemma 3.5. *Consider random variables $\mathbf{X}, \mathbf{X}^1, \ldots, \mathbf{X}^t$ on $\{0,1\}^n$ and $\mathbf{Y}, \mathbf{Y}^1, \ldots, \mathbf{Y}^t$ on $\{0,1\}^m$ with the guarantees that $(\mathbf{X}, \mathbf{X}^1, \ldots, \mathbf{X}^t)$ is independent of $(\mathbf{Y}, \mathbf{Y}^1, \ldots, \mathbf{Y}^t)$ and $\mathbf{Y}$ is uniform. Let $\mathrm{Ext}_x : \{0,1\}^n \times \{0,1\}^{d_y} \to \{0,1\}^{d_x}$ be a (k_x, ε)-strong seeded extractor and $\mathrm{Ext}_y : \{0,1\}^m \times \{0,1\}^{d_x} \to \{0,1\}^{d_y}$ be a (k_y, ε)-strong seeded extractors. In addition, let $\mathbf{S}_0, (\mathbf{S}_0^i)_{i\in[t]}$ be the length-d prefixes of $\mathbf{Y}, (\mathbf{Y}_0^i)_{i\in[t]}$ respectively. Then for every $j \in \mathbb{N}$ define $\mathbf{R}_j, \mathbf{R}_j^i, \mathbf{S}_j, \mathbf{S}_j^i$ recursively as follows:*

$$\begin{aligned}
\mathbf{R}_j &:= \mathrm{Ext}_x(\mathbf{X}, \mathbf{S}_{j-1}), \\
\mathbf{R}_j^i &:= \mathrm{Ext}_x(\mathbf{X}, \mathbf{S}_{j-1}) \ \forall i \in [t], \\
\mathbf{S}_j &:= \mathrm{Ext}_y(\mathbf{Y}, \mathbf{R}_j), \\
\mathbf{S}_j^i &:= \mathrm{Ext}_y(\mathbf{Y}, \mathbf{R}_j) \ \forall i \in [t].
\end{aligned}$$

Then the following claims hold for every $j \in \mathbb{N}$.

- *If $\mathrm{H}_\infty(\mathbf{X}) \geq k_x + \log(1/\varepsilon) + (t+1)(j-1)d_x$, then*

$$\mathbf{R}_j \approx_{(4j-2)\varepsilon} \mathbf{U}_d \mid \left(\mathbf{Y}, \mathbf{Y}^{[t]}, \mathbf{R}_{<j}, \mathbf{R}_{<j}^{[t]}, \mathbf{S}_{<j}, \mathbf{S}_{<j}^{[t]}\right). \tag{3.1}$$

- *If $m \geq k_y + \log(1/\varepsilon) + (t+1)jd_y$, then*

$$\mathbf{S}_j \approx_{4j\varepsilon} \mathbf{U}_d \mid \left(\mathbf{X}, \mathbf{X}^{[t]}, \mathbf{R}_{\leq j}, \mathbf{R}_{\leq j}^{[t]}, \mathbf{S}_{<j}, \mathbf{S}_{<j}^{[t]}\right). \tag{3.2}$$

Proof. We will prove the two claims and the following conditions by induction on j, and we claim that these conditions imply the two claims.

$$\widetilde{\mathrm{H}}_\infty(\mathbf{X} \mid \mathbf{R}_{<j}, \mathbf{R}_{<j}^{[t]}, \mathbf{S}_{<j}, \mathbf{S}_{<j}^{[t]}) \geq \mathrm{H}_\infty(\mathbf{X}) - (t+1)(j-1)d_x \tag{3.3}$$

$$\widetilde{\mathrm{H}}_\infty(\mathbf{Y} \mid \mathbf{R}_{\leq j}, \mathbf{R}_{\leq j}^{[t]}, \mathbf{S}_{<j}, \mathbf{S}_{<j}^{[t]}) \geq m - (t+1)jd_y \tag{3.4}$$

$$(\mathbf{X}, \mathbf{X}^{[t]}) \leftrightarrow (\mathbf{R}_{\leq j}, \mathbf{R}_{\leq j}^{[t]}, \mathbf{S}_{<j}, \mathbf{S}_{<j}^{[t]}) \leftrightarrow (\mathbf{Y}, \mathbf{Y}^{[t]}) \text{ forms a Markov Chain} \tag{3.5}$$

First we start with the base cases. The $j = 0$ base cases for (3.2), (3.4) and (3.5) are trivial. Next observe that $(\mathbf{S}_0, \mathbf{S}_0^{[t]})$ is and independent function of $(\mathbf{Y}, \mathbf{Y}^{[t]})$, and thus is independent of $\mathbf{X}$. Therefore the $j = 1$ base case for (3.3) is also true, because $\widetilde{\mathrm{H}}_\infty(\mathbf{X} \mid \mathbf{S}_0, \mathbf{S}_0^{[t]}) = \mathrm{H}_\infty(\mathbf{X})$. In addition, observe that $\mathbf{S}_0$ is uniform and independent of $\mathbf{X}$. By Lemma 2.34, (3.1) is also true for the $j = 1$ case.

Next we argue the inductive case in a similar order. For $\ell \in \mathbb{N}$, assume that (3.2), (3.4) and (3.5) are true for $j < \ell$, and that (3.3) and (3.1) are true for $j \leq \ell$. First of all, observe that because $(\mathbf{S}_{\ell-1}, \mathbf{S}_{\ell-1}^{[t]})$ is a deterministic function of $(\mathbf{Y}, \mathbf{Y}^{[t]})$ (and $(\mathbf{R}_{\ell-1}, \mathbf{R}_{\ell-1}^{[t]})$ if $\ell > 1$), by further conditioning on $(\mathbf{S}_{\ell-1}, \mathbf{S}_{\ell-1}^{[t]})$ we see that (3.5) for $j = \ell - 1$ implies

$$(\mathbf{X}, \mathbf{X}^{[t]}) \leftrightarrow (\mathbf{R}_{<\ell}, \mathbf{R}_{<\ell}^{[t]}, \mathbf{S}_{<\ell}, \mathbf{S}_{<\ell}^{[t]}) \leftrightarrow (\mathbf{Y}, \mathbf{Y}^{[t]}).$$

In addition, because $(\mathbf{S}_{\ell-1}, \mathbf{S}_{\ell-1}^{[t]})$ has length $(t+1)d_y$, (3.4) for $j = \ell - 1$ and the chain rule for min-entropy (Lemma 2.10) also implies

$$\widetilde{\mathrm{H}}_\infty(\mathbf{Y} \mid \mathbf{R}_{<\ell}, \mathbf{R}_{<\ell}^{[t]}, \mathbf{S}_{<\ell}, \mathbf{S}_{<\ell}^{[t]}) \geq m - (t+1)\ell d.$$

After conditioning on $(\mathbf{S}_{\ell-1}, \mathbf{S}_{\ell-1}^{[t]})$, now $(\mathbf{R}_\ell, \mathbf{R}_\ell^{[t]})$ becomes a deterministic function of $(\mathbf{X}, \mathbf{X}^{[t]})$ and is independent of $(\mathbf{Y}, \mathbf{Y}^{[t]})$. Therefore if we further condition on $(\mathbf{R}_\ell, \mathbf{R}_\ell^{[t]})$ we can conclude that (3.4) and (3.5) are both true for $j = \ell$. And these two conditions together with (3.1) for $j = \ell$ would imply (3.2) by Lemma 2.34. Furthermore, because $(\mathbf{R}_\ell, \mathbf{R}_\ell^{[t]})$ has length $(t+1)d_x$, (3.3) for $j = \ell$ would imply that $\widetilde{\mathrm{H}}_\infty(\mathbf{X} \mid \mathbf{R}_{\leq\ell}, \mathbf{R}_{\leq\ell}^{[t]}, \mathbf{S}_{<\ell}, \mathbf{S}_{<\ell}^{[t]}) \geq \mathrm{H}_\infty(\mathbf{X}) - (t+1)\ell d$. If we further condition on $(\mathbf{S}_\ell, \mathbf{S}_\ell^{[t]})$, which is now independent of $\mathbf{X}$, we can conclude that (3.3) is also true for $j = \ell + 1$, and $(\mathbf{X}, \mathbf{X}^{[t]}) \leftrightarrow (\mathbf{R}_{\leq\ell}, \mathbf{R}_{\leq\ell}^{[t]}, \mathbf{S}_{\leq\ell}, \mathbf{S}_{\leq\ell}^{[t]}) \leftrightarrow (\mathbf{Y}, \mathbf{Y}^{[t]})$. Finally, by (3.3) for $j = \ell + 1$ and (3.2) for $j = \ell$ we can conclude that (3.1) is also true for $j = \ell + 1$. □

Note that in the proof above, we are always maintaining a Markov chain

$(\mathbf{X}, \mathbf{X}^{[t]}, \mathbf{R}_j, \mathbf{R}_j^{[t]}) \leftrightarrow \mathbf{Z}_i \leftrightarrow (\mathbf{Y}, \mathbf{Y}^{[t]})$, where $\mathbf{Z}_j = (\mathbf{R}_{\le j}, \mathbf{R}_{\le j}^{[t]}, \mathbf{S}_{<j}, \mathbf{S}_{<j}^{[t]})$. This make sure that we can use $\mathbf{R}_j$ as a seed to extract from $\mathbf{Y}$ and repeat the argument for another round. This will be an important feature for the construction of correlation breakers: we will always consider a Markov chain $(\mathbf{X}, \mathbf{X}^{[t]}, \mathbf{R}, \mathbf{R}^{[t]}) \leftrightarrow (\mathbf{Z}) \leftrightarrow (\mathbf{Y}, \mathbf{Y}^{[t]})$, where $\mathbf{X}, \mathbf{Y}$ still have some entropy after conditioning on $\mathbf{Z}$, and $\mathbf{R}$ is close to uniform. Then we will use $\mathbf{R}$ as a seed to compute some "strong function" $\mathbf{S} = f(\mathbf{Y}, \mathbf{R})$ so that $\mathbf{S}$ satisfies certain properties even after conditioning on $\mathbf{R}$ (and its tampered versions). Then the Markov chain we consider in the next round will become $(\mathbf{X}, \mathbf{X}^{[t]}) \leftrightarrow \mathbf{Z} \leftrightarrow (\mathbf{Y}, \mathbf{Y}^{[t]}, \mathbf{S}, \mathbf{S}^{[t]})$. Note that the alternating extraction process itself is also a strong function if we consider the output to be $\mathbf{R}_1, \dots, \mathbf{R}_j$. This is called a look-ahead extractor in [DW09].

Lemma 3.6 (ℓ-look-ahead extractor [DW09])**.** *Fix parameters $n, m, t, \ell \in \mathbb{N}$ and $\varepsilon > 0$. There exists an explicit function $\mathrm{laExt}_\ell : \{0,1\}^n \times \{0,1\}^d \to (\{0,1\}^m)^\ell$ which satisfies the following. Let $\mathbf{X}, \mathbf{X}^{[t]} \in \{0,1\}^n$ and $\mathbf{Y}, \mathbf{Y}^{[t]} \in \{0,1\}^d$ be random variables such that $(\mathbf{X}, \mathbf{X}^{[t]})$ is independent of $(\mathbf{Y}, \mathbf{Y}^{[t]})$, $\mathbf{Y} = \mathbf{U}_d$ and $\mathrm{H}_\infty(\mathbf{X}) \ge k$ for some parameter k to be specified later. Then $(\mathbf{R}_1, \dots, \mathbf{R}_\ell)$ and their tampered versions $(\mathbf{R}_1^{[t]}, \dots, \mathbf{R}_\ell^{[t]})$ satisfy*

$$\forall i \in [\ell] \ (\mathbf{R}_i \approx_\varepsilon \mathbf{U}_m) \mid (\mathbf{Y}, \mathbf{Y}^{[t]}, \mathbf{R}_{<i}, \mathbf{R}_{<i}^{[t]}).$$

In addition, it suffices to take $d = O(t\ell \log(n\ell/\varepsilon))$ and $k = O(t\ell \log(d\ell/\varepsilon))$.

Proof. Simply substitute explicit extractors from Lemma 2.17 with error $\varepsilon/4\ell$ into Lemma 3.5. □

A look-ahead extractor can already give us a "semi-correlation breaker" that will work with the additional guarantee that $\alpha > \alpha^i$ for every i. Specifically, consider $A = 2^a$ and define $(\mathbf{R}_1, \dots, \mathbf{R}_A) := \mathrm{laExt}_A(\mathbf{X}, \mathbf{S})$, then we can take $\mathrm{CB}(\mathbf{X}, \mathbf{S}, \alpha) = \mathbf{R}_\alpha$. (Here we view the a-bit advice as an integer from $[A]$). In fact, a semi-correlation

breaker suffices in the construction of two-source extractors, and the very first construction of correlation breakers in [Li13] is in fact a semi-correlation breaker. However, the parameter here is terrible: we need the entropy k to be $O(Am)$, but in the construction we discuss in Section 3.2, A needs to be 2^d where d is the seed length of a seeded extractor, and this implies $A = \mathrm{poly}(n)$. Obviously it is not even possible to get a n-bit source with entropy $\mathrm{poly}(n)$, and our goal is to extract from entropy $k = o(n)$. In the next section, we will introduce a more efficient way to utilize the advice.

3.4 Merging Independence

To avoid the trivial $O(2^a)$ dependence of entropy on the advice length a, Li [Li13] observed an important property that the output of a seeded extractor can inherit the independence of the seed from its tampered version. That is, if $\mathbf{S}$ is uniform conditioned on $\mathbf{S}'$, then $\mathrm{Ext}(\mathbf{X}, \mathbf{S})$ is also uniform conditioned on $\mathrm{Ext}(\mathbf{X}, \mathbf{S}')$. To see why this is true, first we condition on $\mathbf{S}'$ and then on $\mathrm{Ext}(\mathbf{X}, \mathbf{S}')$. By chain rule this decreases the entropy of $\mathbf{X}$ by roughly the length of $\mathrm{Ext}(\mathbf{X}, \mathbf{S}')$. Then it remains true that $\mathbf{S}$ is uniform, $\mathbf{X}$ has enough entropy and $\mathbf{X}$ is independent of $\mathbf{S}$. Therefore we can conclude that $\mathrm{Ext}(\mathbf{X}, \mathbf{S})$ is uniform even when conditioned on $\mathbf{S}$. In fact, by a similar argument we can also prove (with ideas from a work by Chattopadhyay and Li [CL16a]) that strong seeded extractor inherit the independence of the source from its tampered version. We call this the "independence-merging property" of a strong seeded extractor.

Lemma 3.7 (independence-merging lemma)**.** *Let* $(\mathbf{X}, \mathbf{X}^{[t]}) \leftrightarrow \mathbf{Z} \leftrightarrow (\mathbf{Y}, \mathbf{Y}^{[t]})$ *be a Markov chain, such that* $\mathbf{X}, \mathbf{X}^{[t]} \in \{0,1\}^n$, $\mathbf{Y}, \mathbf{Y}^{[t]} \in \{0,1\}^d$. *Moreover, suppose there exists* $S, T \subseteq [t]$ *such that*

- $(\mathbf{Y} \approx_\delta \mathbf{U}_d) \mid (\mathbf{Z}, \mathbf{Y}^S)$
- $\widetilde{\mathrm{H}}_\infty(\mathbf{X} \mid (\mathbf{X}^T, \mathbf{Z})) \geq k + tm + \log(1/\varepsilon)$

Let $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *be any* (k, ε)*-strong seeded extractor,* $\mathbf{W} := \mathrm{Ext}(\mathbf{X}, \mathbf{Y})$ *and* $\mathbf{W}^j := \mathrm{Ext}(\mathbf{X}^j, \mathbf{Y}^j)$ *for every* $j \in [t]$*. Then*

$$(\mathbf{W} \approx_{2\varepsilon+\delta} \mathbf{U}_m) \mid (\mathbf{W}^{S \cup T}, \mathbf{Y}, \mathbf{Y}^{[t]}, \mathbf{Z}).$$

Proof. Without loss of generality, assume that S and T are disjoint. First we give a short sentence as a sketch of the proof: fix $\mathbf{X}^T, \mathbf{Y}^S$ and $\mathbf{W}^S$, observe that $\mathbf{X}$ still have enough entropy and $\mathbf{Y}$ is still uniform and independent of $\mathbf{X}$. Therefore $\mathbf{W} = \mathrm{Ext}(\mathbf{X}, \mathbf{Y})$ is still uniform after fixing $\mathbf{Y}$ and $\mathbf{Y}^{[t]}$. Now $\mathbf{W}^T$ is also fixed, so we can conclude that $\mathbf{W}$ is uniform conditioned on $\mathbf{Z}, \mathbf{Y}, \mathbf{Y}^{[t]}, \mathbf{W}^{S \cup T}$. The formal proof is as follows. First note that $\mathbf{Y}$ is δ-close to uniform conditioned on $\mathbf{Z}, \mathbf{Y}^S$, and will still be δ-close to uniform even when further conditioned on $\mathbf{X}^S, \mathbf{X}^T$. That is,

$$(\mathbf{Y} \approx_\delta \mathbf{U}_d) \mid (\mathbf{X}^T, \mathbf{Z}, \mathbf{X}^S, \mathbf{Y}^S).$$

Since $\mathbf{W}^S$ is a deterministic function of $\mathbf{X}^S, \mathbf{Y}^S$, this implies

$$(\mathbf{Y} \approx_\delta \mathbf{U}_d) \mid (\mathbf{X}^T, \mathbf{Z}, \mathbf{W}^S, \mathbf{Y}^S).$$

In addition, by chain rule (Lemma 2.10),

$$\widetilde{\mathrm{H}}_\infty(\mathbf{X} \mid (\mathbf{X}^T, \mathbf{Z}, \mathbf{W}^S)) \geq k + (t - |S|)m + \log(1/\varepsilon) \geq k + \log(1/\varepsilon).$$

This implies

$$\widetilde{\mathrm{H}}_\infty(\mathbf{X} \mid (\mathbf{X}^T, \mathbf{Z}, \mathbf{W}^S, \mathbf{Y}^S)) \geq k + \log(1/\varepsilon)$$

since $\mathbf{X} \leftrightarrow \mathbf{Z} \leftrightarrow \mathbf{Y}^S$ is a Markov chain. Next observe that by fixing $\mathbf{X}^T$ and $\mathbf{Y}^S$ in the Markov chain $(\mathbf{X}, \mathbf{X}^{[t]}) \leftrightarrow \mathbf{Z} \leftrightarrow (\mathbf{Y}, \mathbf{Y}^{[t]})$, we get that $(\mathbf{X}, \mathbf{X}^S) \leftrightarrow (\mathbf{X}^T, \mathbf{Z}, \mathbf{Y}^S) \leftrightarrow (\mathbf{Y}, \mathbf{Y}^{[t]})$. Since $\mathbf{W}^S = (\mathrm{Ext}(\mathbf{X}^i, \mathbf{Y}^i))_{i \in S}$, by fixing $\mathbf{W}^S$ we get a Markov chain

$$\mathbf{X} \leftrightarrow (\mathbf{X}^T, \mathbf{Z}, \mathbf{W}^S, \mathbf{Y}^S) \leftrightarrow (\mathbf{Y}, \mathbf{Y}^{[t]}).$$

Therefore Lemma 2.34 implies

$$(\mathbf{W} \approx_{2\varepsilon+\delta} \mathbf{U}_m) \mid (\mathbf{X}^T, \mathbf{Z}, \mathbf{W}^S, \mathbf{Y}^S, \mathbf{Y}, \mathbf{Y}^{[t]}).$$

Now note that $\mathbf{W}^T$ is a deterministic function of $\mathbf{X}^T$ and $\mathbf{Y}^T$, so

$$(\mathbf{W} \approx_{2\varepsilon+\delta} \mathbf{U}_m) \mid (\mathbf{W}^T, \mathbf{Z}, \mathbf{W}^S, \mathbf{Y}^S, \mathbf{Y}, \mathbf{Y}^{[t]})$$

By removing duplicated $\mathbf{Y}^S$ we prove the claim. □

It is not hard to prove that the independence-merging property also holds for look-ahead extractors. Given this property, the idea in [Li13] is as follows. Note that for a one-bit advice, the construction in the previous section is not too costly: we can take $(\mathbf{R}_0, \mathbf{R}_1) := \mathrm{laExt}_2(\mathbf{X}, \mathbf{S})$, and define $\mathrm{CB}(\mathbf{X}, \mathbf{S}, \alpha) := \mathrm{laExt}_2(\mathbf{X}, \mathbf{S})$. Given a-bit advice string $\alpha = (\alpha_1, \ldots, \alpha_a)$, to compute $\mathrm{CB}(\mathbf{X}, \mathbf{Y}, \alpha)$, we can first take $\mathbf{S}$ to be a prefix of $\mathbf{Y}$ (which is also uniform), then define $(\mathbf{R}_0, \mathbf{R}_1) := \mathrm{laExt}_2(\mathbf{X}, \mathbf{S})$, then update $\mathbf{S}$ to be $\mathrm{Ext}(\mathbf{Y}, \mathbf{R}_{\alpha_1})$ so that it is again independent of $\mathbf{X}$. Then we recompute $(\mathbf{R}_0, \mathbf{R}_1) := \mathrm{laExt}_2(\mathbf{X}, \mathbf{S})$ and update $\mathbf{S}$ to be $\mathrm{Ext}(\mathbf{Y}, \mathbf{R}_{\alpha_2})$, and repeat until we go through every bit of α. If $\alpha > \alpha^i$ then at some point we will get the guarantee that $\alpha_j > \alpha^i_j$, and the 2-look-ahead extractor will guarantee independence from the tampered version. Our final $\mathbf{S}$ inherits such independence at some point and thus is also uniform conditioned on the tampered version. To deal with the case that $\alpha < \alpha^i$, we can use another idea by Cohen [Coh16a] called "flip-flop": simply flip each bit of $\alpha, \alpha^{[t]}$ and repeat the process for another round. If $\alpha < \alpha^i$ then in the second round the advice strings become $\neg\alpha > \neg\alpha^i$, where $\neg$ denotes bitwise negation. To summarize, the idea here is to consider each advice bit individually and get the guarantee that independence is generated somewhere. Then the independence-merging property of strong-seeded extractors ensures that we can *sequentially* merge the independence. The dependence of entropy on the advice length now becomes $O(a)$.

To get an even better dependence on the advice length, [CL16a] showed that we can merge the independence *in parallel*. To see how it works, we again take $(\mathbf{R}_0, \mathbf{R}_1) := \mathrm{laExt}_2(\mathbf{X}, \mathbf{S})$, then consider the sequence

$$(\mathbf{V}_1, \mathbf{V}_2, \ldots, \mathbf{V}_{2a}) := (\mathbf{R}_{\alpha_1}, \mathbf{R}_{\alpha_2}, \ldots, \mathbf{R}_{\alpha_a}, \mathbf{R}_{1-\alpha_1}, \ldots, \mathbf{R}_{1-\alpha_a}).$$

As we discussed in the last paragraph, for the i-th tampered version with advice $\alpha^i \neq \alpha$, there exists some j such that $\mathbf{V}_j$ is uniform conditioned on its tampered version $\mathbf{V}_j^i$. Our goal is to merge the whole sequence into one random variable so that the final output inherits the independence from all random variables. If we sequentially run alternating extraction between $\mathbf{Y}$ and every $\mathbf{V}_j$ to "collect independence" from every $\mathbf{V}_j$, the dependence on the advice length is again $O(a)$. However, it turns out that one can use constant round of alternating extraction to merge all pairs of adjacent random variables $(\mathbf{V}_1, \mathbf{V}_2), (\mathbf{V}_3, \mathbf{V}_4), \ldots, (\mathbf{V}_{2a-1}, \mathbf{V}_{2a})$ *in parallel*. If we repeat this process for $\log(2a)$ round, then we can get a correlation breaker with an entropy requirement that is only proportional to $O(\log(a))$.

To complete this merging task, we need a primitive called *independence-preserving merger* (IPM for short), which was first introduce by Cohen and Schulman [CS16a] and then generalized by Chattopadhyay and Li [CL16a]. This primitive takes two random variables $\mathbf{V}_1, \mathbf{V}_2$ that have enough entropy, and merge them into one uniform random variable that preserves the independence of both $\mathbf{V}_1, \mathbf{V}_2$ from their tampered versions, with the help of a uniform seed $\mathbf{S}$. Briefly speaking, to construct a IPM, we simply run alternating extraction between the seed $\mathbf{S}$ and $\mathbf{V}_1, \mathbf{V}_2$, and use the independence-merging lemma to argue that the independence of $\mathbf{V}_1, \mathbf{V}_2$ from their tampered versions are preserved. In order to make sure that the final output has the same length as $\mathbf{V}_1$ and $\mathbf{V}_2$ (so that we can repeat the merging process for more rounds), we let the IPM takes an additional "entropy pool" $\mathbf{Y}$ as input, and in the last step we use the output of alternating extraction as a seed to extract from $\mathbf{Y}$

and get a random variable that has the same length as $\mathbf{V}_1$.

Lemma 3.8 (2-IPM). *Fix parameters $r, n \in \mathbb{N}$ and $\varepsilon > 0$. There exists $s = O(t \log(n/\varepsilon))$, $k = O(tr + t\log(n/\varepsilon))$ and an explicit function $\mathrm{IPM}_2 : (\{0,1\}^r)^2 \times \{0,1\}^n \times \{0,1\}^s \to \{0,1\}^r$ which satisfies the following. Consider random variables $\mathbf{V}_1, \mathbf{V}_2, \mathbf{V}_1^{[t]}, \mathbf{V}_2^{[t]} \in \{0,1\}^r$,$\mathbf{S}, \mathbf{S}^{[t]} \in \{0,1\}^s$,$\mathbf{Y}, \mathbf{Y}^{[t]} \in \{0,1\}^n$ that satisfy the following conditions for some $T_1, T_2 \subseteq [t]$:*

- $(\mathbf{S}, \mathbf{S}^{[t]})$ *and* $(\mathbf{V}_1, \mathbf{V}_1^{[t]}, \mathbf{V}_2, \mathbf{V}_2^{[t]}, \mathbf{Y}, \mathbf{Y}^{[t]})$ *are independent*
- $\widetilde{\mathrm{H}}_\infty(\mathbf{V}_1 \mid \mathbf{V}_1^{T_1}), \widetilde{\mathrm{H}}_\infty(\mathbf{V}_2 \mid \mathbf{V}_2^{T_2}) \geq s$
- $\mathbf{S}$ *is uniform*
- $\mathrm{H}_\infty(\mathbf{Y}) \geq k$

Then $\mathbf{W} = \mathrm{IPM}_2((\mathbf{V}_1, \mathbf{V}_2), \mathbf{Y}, \mathbf{S})$ and its tampered versions $\mathbf{W}^{[t]}$ satisfy

$$(\mathbf{W} \approx_{O(\varepsilon)} \mathbf{U}_r) \mid (\mathbf{W}^{T_1 \cup T_2}, \mathbf{S}, \mathbf{S}^{[t]}).$$

Proof. This function requires the following extractors, and from Lemma 2.17 we can see it suffices to take $d = O(\log(n/\varepsilon))$.

- $\mathrm{Ext}_V : \{0,1\}^r \times \{0,1\}^d \to \{0,1\}^d$ is a $(2d, \varepsilon)$-strong seeded extractor
- $\mathrm{Ext}_S : \{0,1\}^s \times \{0,1\}^d \to \{0,1\}^d$ is a $(2d, \varepsilon)$-strong seeded extractor
- $\mathrm{Ext}_X : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^n$ is a $(2r, \varepsilon)$-strong seeded extractor

The construction of IPM_2 is as follows. Let $\mathbf{Q}_1$ be a length-d prefix of $\mathbf{S}$. Now apply

the following steps:

$$\begin{aligned}
\mathbf{P}_1 &= \mathrm{Ext}_V(\mathbf{V}_1, \mathbf{Q}_1) \\
\mathbf{Q}_2 &= \mathrm{Ext}_S(\mathbf{S}, \mathbf{P}_1) \\
\mathbf{P}_2 &= \mathrm{Ext}_V(\mathbf{V}_2, \mathbf{Q}_1) \\
\mathbf{Q}_r &= \mathrm{Ext}_S(\mathbf{S}, \mathbf{P}_2) \\
\mathbf{W} &= \mathrm{Ext}_X(\mathbf{Y}, \mathbf{Q}_r).
\end{aligned}$$

Then $\mathbf{W}$ will be the output of $\mathrm{IPM}_2((\mathbf{V}_1, \mathbf{V}_2), \mathbf{Y}, \mathbf{S})$. To prove the correctness, we fix $(\mathbf{Q}_1, \mathbf{Q}_1^{[t]})$, $(\mathbf{P}_1, \mathbf{P}_1^{[t]})$, $(\mathbf{Q}_2, \mathbf{Q}_2^{[t]})$ and $(\mathbf{P}_2, \mathbf{P}_2^{[t]})$ in order. Observe that as long as the extractor applied in each step satisfies the conditional min-entropy requirement in Lemma 3.7, by applying Lemma 3.7 in each step we get

$$\begin{aligned}
(\mathbf{P}_1 \approx_{\delta} \mathbf{U}_d) \mid (\mathbf{P}_1^{T_1} &, \mathbf{Q}_1, \mathbf{Q}_1^{[t]}) \\
(\mathbf{Q}_2 \approx_{\delta+2\varepsilon} \mathbf{U}_d) \mid (\mathbf{Q}_2^{T} &, \mathbf{Q}_1, \mathbf{Q}_1^{[t]}, \mathbf{P}_1, \mathbf{P}_1^{[t]}) \\
(\mathbf{P}_2 \approx_{\delta+4\varepsilon} \mathbf{U}_d) \mid (\mathbf{P}_2^{T_1 \cup T_2} &, \mathbf{Q}_1, \mathbf{Q}_1^{[t]}, \mathbf{P}_1, \mathbf{P}_1^{[t]}, \mathbf{Q}_2, \mathbf{Q}_2^{[t]}) \\
(\mathbf{Q}_r \approx_{\delta+6\varepsilon} \mathbf{U}_d) \mid (\mathbf{Q}_r^{T_1 \cup T_2} &, \mathbf{Q}_1, \mathbf{Q}_1^{[t]}, \mathbf{P}_1, \mathbf{P}_1^{[t]}, \mathbf{Q}_2, \mathbf{Q}_2^{[t]}, \mathbf{P}_2, \mathbf{P}_2^{[t]},) \\
(\mathbf{W} \approx_{\delta+8\varepsilon} \mathbf{U}_m) \mid (\mathbf{W}^{T_1 \cup T_2} &, \mathbf{Q}_1, \mathbf{Q}_1^{[t]}, \mathbf{P}_1, \mathbf{P}_1^{[t]}, \mathbf{Q}_2, \mathbf{Q}_2^{[t]}, \mathbf{P}_2, \mathbf{P}_2^{[t]}, \mathbf{Q}_r, \mathbf{Q}_r^{[t]}).
\end{aligned} \tag{3.6}$$

Furthermore, observe that by further fixing $(\mathbf{Q}_r, \mathbf{Q}_r^{[t]}, \mathbf{W}^{T_1 \cup T_2})$ in the last step we get

$$(\mathbf{S}, \mathbf{S}^{[t]}) \leftrightarrow (\mathbf{Q}_1, \mathbf{Q}_1^{[t]}, \mathbf{P}_1, \mathbf{P}_1^{[t]}, \mathbf{Q}_2, \mathbf{Q}_2^{[t]}, \mathbf{P}_2, \mathbf{P}_2^{[t]}, \mathbf{Q}_r, \mathbf{Q}_r^{[t]}, \mathbf{W}^{T_1 \cup T_2}) \leftrightarrow \mathbf{W}.$$

Therefore we can take (3.6) and further condition on $(\mathbf{S}, \mathbf{S}^{[t]})$ to prove the claim. Finally we need to verify that at each step the conditional entropy of $\mathbf{V}_1, \mathbf{V}_2, \mathbf{S}, \mathbf{Y}$ actually satisfies the requirement in Lemma 3.7. To simplify notation, we define

$\mathbf{Z} = (\mathbf{Q}_1^{[t]}, \mathbf{Q}_1), (\mathbf{P}_1^{[t]}, \mathbf{P}_1), (\mathbf{Q}_2^{[t]}, \mathbf{Q}_2)$ for short. We need to make sure that

$$\begin{aligned}
\widetilde{\mathrm{H}}_\infty(\mathbf{V}_1 \mid \mathbf{Z}, \mathbf{V}_1^{T_1}) &\geq \widetilde{\mathrm{H}}_\infty(\mathbf{V}_1 \mid \mathbf{V}_1^{T_1}) - (t+1)d \geq 2d + td + \log(1/\varepsilon) \\
\widetilde{\mathrm{H}}_\infty(\mathbf{V}_2 \mid \mathbf{Z}, \mathbf{V}_2^{T_2}) &\geq \widetilde{\mathrm{H}}_\infty(\mathbf{V}_2 \mid \mathbf{V}_2^{T_2}) - (t+1)d \geq 2d + td + \log(1/\varepsilon) \\
\widetilde{\mathrm{H}}_\infty(\mathbf{S} \mid \mathbf{Z}) &\geq \mathrm{H}_\infty(\mathbf{S}) - 2(t+1)d \qquad\quad \geq 2d + td + \log(1/\varepsilon) \\
\widetilde{\mathrm{H}}_\infty(\mathbf{Y} \mid \mathbf{Z}, \mathbf{P}_2^{[t]}, \mathbf{P}_2) &\geq \mathrm{H}_\infty(\mathbf{Y}) - 2(t+1)d \qquad\quad \geq 2r + tr + \log(1/\varepsilon).
\end{aligned}$$

It suffices to take $s = O(td) = O(t \log(n/\varepsilon))$ and $k = O(tr + td) = O(tr + t \log(n/\varepsilon))$.

□

With the help of IPM, we get a construction of correlation breakers $\mathrm{CB}(\mathbf{X}, \mathbf{Y}, \alpha)$ that works as follows. (Note that the construction here can be viewed as a variant of the construction in [Li17]. There are other more efficient but more complicated constructions in [Li19, Li23], which we will not discuss in details in this thesis.)

1. First take a prefix of $\mathbf{Y}$, denoted by $\mathbf{Y}_0$, and compute $(\mathbf{R}_1, \mathbf{R}_0) = \mathrm{laExt}_2(\mathbf{X}, \mathbf{Y}_0)$.
2. Assign $(\mathbf{V}_1, \mathbf{V}_2, \ldots, \mathbf{V}_{2a}) = (\mathbf{R}_{\alpha_1}, \mathbf{R}_{\alpha_2}, \ldots, \mathbf{R}_{\alpha_a}, \mathbf{R}_{1-\alpha_1}, \ldots, \mathbf{R}_{1-\alpha_a})$.
3. Take a prefix of $\mathbf{V}_1$, denoted by $\mathbf{Q}$, to extract a uniform string $\mathbf{S} := \mathrm{Ext}(\mathbf{X}, \mathbf{Q})$, which will be used as a seed for the IPM_2. Note that after conditioning on $\mathbf{Q}$ and their tampered versions, each $\mathbf{V}_j$ still has enough entropy.
4. Use $\mathbf{S}$ as the seed and $\mathbf{Y}$ as the entropy pool to merge every two random variables with IPM_2. Repeat from the previous step until there is only one string $\mathbf{V}_1$ left.
5. Use the final $\mathbf{V}_1$ as the seed to extract a long enough output from $\mathbf{X}$.

3.5 BDT Error Reduction

Finally we briefly discuss the parameters of the (q, t)-NOBF source we get, and a barrier we will face when applying majority as the NOBF extractor in the last step. Recall that in Section 3.2, we take a strong seeded extractor $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ with seed length d and enumerate all $D = 2^d$ seeds to get a sequence of random variables $(\mathbf{Y}_1, \ldots, \mathbf{Y}_D)$ such that only ε fraction of the random variables are not uniform. Then we apply a correlation breaker to get a $(\varepsilon D, t)$-NOBF source. If we choose to apply the Chattopadhyay-Zuckerman NOBF extractor, then we need $t = \mathrm{polylog}(n)$. Recall that in the alternating extraction process (Lemma 3.5) we need the sources to have at least $t \log(1/\varepsilon)$ entropy. Therefore, the corresponding two-source extractors also require $\mathrm{polylog}(n)$ entropy. If we hope to get an extractor with entropy requirement as low as $O(\log^{1+o(1)}(n))$, then we need to apply the majority function as the NOBF-source extractor, which only require $t = O(1)$. However, in this case we need $\varepsilon D \ll \sqrt{D}$, which turns out to be a tricky requirement: the best known seeded extractors (e.g. [GUV09]) require $D = (n/\varepsilon)^C$ for some large constant C, which implies $\varepsilon D = D^{1-\gamma}$ for some really small $\gamma > 0$. In fact, as pointed out by Cohen and Schulman [CS16a], one cannot hope to get $\varepsilon D < \sqrt{D}$ because of the $D > (1/\varepsilon)^2$ lower bound for seeded extractors [RT00].

To bypass this barrier, Ben-Aroya, Doron and Ta-Shma [BDT19] showed a modified construction of NOBF sources based on an error reduction framework (which we call the "BDT error reduction" throughout this thesis). The intuition is as follows. In the original sequence $(\mathbf{Y}_1, \ldots, \mathbf{Y}_D)$, if we randomly pick one index i from $[D]$, the probability that $\mathbf{Y}_i$ is uniform is only $1 - \varepsilon$, where ε is not small enough to satisfy $\varepsilon D \ll \sqrt{D}$. To reduce the error, observe that if we use some randomness r to sample A indices $i_1, \ldots, i_A$ independently, then with probability at least $1 - \varepsilon^A$

we have at least one $\mathbf{Y}_{i_j}$ that is uniform. While we do not know exactly which index i_j is uniform, we know that the corresponding correlation breaker output $\mathbf{Z}_{i_j}$ is uniform conditioned on other $\{\mathbf{Z}_{i_{j'}}\}_{j \neq j}$, as long as the independence number t is greater than A. Therefore, we can generate one uniform bit without knowing i_j by taking $\mathbf{Z}'_r := \mathbf{Z}_{i_1} \oplus \ldots \oplus \mathbf{Z}_{i_A}$. (In other words, a correlation breaker can be used to construct a merger, as observed by Cohen [Coh16a].) If we enumerate all the possible randomness r we get an NOBF source $(\mathbf{Z}'_r)_r$ such that only $\varepsilon' = \varepsilon^A \ll \varepsilon$ fraction of the bits are not uniform.

Note that there are $D' = D^A$ possible choices of r, so we are not getting any improvement because we still have $\varepsilon' D' > \sqrt{D'}$. However, if we use a randomness-efficient way to sample $i_1, \ldots, i_A$, then we might be able to get $\varepsilon' D' \ll \sqrt{D'}$. To complete this task, observe that our goal is to sample multiple indices $i_1, \ldots, i_A$ so that the probability that every i_j falls into the bad ε fraction of indices is much smaller than ε. This is exactly what a *disperser* does. If we take the disperser by Zuckerman [Zuc07] (Lemma 2.21) to sample indices, then as long as the starting error is $\varepsilon = 1/\operatorname{poly}(n)$, we can get small enough ε' so that $\varepsilon' D' \leq (D')^{0.4}$, with only $A = O(1)$ samples. We leave the formal statements to Chapter 4 and Chapter 5.

Chapter 4

Affine Extractors for Almost Logarithmic Entropy

In this chapter, we show how to construct an affine extractor that can work for source with min-entropy $\tilde{O}(\log(n))$. The main technical contribution in this chapter is a new linear seeded extractor that has good parameters for short output length, a new construction of affine correlation breakers and a black-box reduction from affine correlation breakers to standard correlation breakers.

This chapter is based on the following joint works:

- [CGL21] Eshan Chattopadhyay, Jesse Goodman, and Jyun-Jie Liao. Affine extractors for almost logarithmic entropy. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021*, pages 622–633. IEEE, 2021
- [CL22] Eshan Chattopadhyay and Jyun-Jie Liao. Extractors for sum of two sources. In *STOC '22: 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1584–1597. ACM, 2022

4.1 Introduction

Two-source extractors and affine extractors are two of the most well-studied models of deterministic extractors. Since the work of Chattopadhyay and Zuckerman [CZ19] which showed how to construct a two-source extractors for $\mathrm{polylog}(n)$ min-entropy, there is a fruitful line of works [BDT19, Coh17, Li17, Li19] that improve the entropy requirement of two-source extractors to $\tilde{O}(\log(n))$. For affine sources, Li [Li16] also showed how to adapt the [CZ19] construction to get an affine extractor for $\mathrm{polylog}(n)$ min-entropy, but it was not known whether one can construct an affine extractor for $\tilde{O}(\log(n))$ before our work ([CGL21]). In this chapter we show the following result.

Theorem 4.1 ([CGL21, CL22])**.** *For any constant $\varepsilon > 0$, there exists a constant $C > 0$ such that for all $n, k \in \mathbb{N}$ such that $k \leq n$, there exists an efficient construction of an extractor $\mathrm{AffExt} : \mathbb{F}_2^n \to \{0,1\}$ with error ε for affine sources with min-entropy $k \geq C \cdot \log n \cdot \log\log n \cdot (\log\log\log n)^3$.*

4.1.1 Proof Overview

Our construction follows the Chattopadhyay-Zuckerman framework introduced in Chapter 3, as in the prior work by Li [Li16]. Recall that in the CZ framework for two-source extractors, we take a strong seeded extractor Ext, apply it on one of the sources $\mathbf{Y}$, enumerate all the seeds and compute a sequence of random variables $(\mathrm{Ext}(\mathbf{Y}, 1), \ldots, \mathrm{Ext}(\mathbf{Y}, D))$. Then we use the second source $\mathbf{X}$ to break the correlation between the random variables. When moving to the setting of affine sources, we do not have access to the second independent source. However, it turns out that one can create an "implicit independent source" by the following lemma.

Lemma 4.2 ([Rao09, Li11]). *Let $\mathbf{X}$ be any random variable sampled from a (n,k)-affine source, and let $L : \{0,1\}^n \to \{0,1\}^m$ be any linear function. Then there exist random variables $\mathbf{A}, \mathbf{B}$ such that $\mathbf{X} = \mathbf{A} + \mathbf{B}$, $\mathrm{H}_\infty(\mathbf{A}) \geq k - m$ and $\mathbf{A}$ is independent of $(L(\mathbf{X}), \mathbf{B})$.*

Suppose we are given an affine source $\mathbf{X}$. If we consider a *linear* strong seeded extractor LExt, then for any seed i we have that $\mathbf{X}$ can be written as $(\mathbf{A}+\mathbf{B})$ where $\mathbf{A}$ is independent of $(\mathbf{B}, \mathrm{LExt}(\mathbf{X}, i))$. Based on this observation, Li [Li16] introduced a primitive called *affine correlation breaker* that can break correlations using this implicit independent source $\mathbf{A}$. The formal definition is as follows.

Definition 4.3. *We say a function $\mathrm{ACB} : \{0,1\}^n \times \{0,1\}^d \times \{0,1\}^a \to \{0,1\}^m$ is a (t,k,ε)-affine correlation breaker if the following holds. Let*

- *$\mathbf{A}, \mathbf{B}$ be random variables on $\{0,1\}^n$ and $\mathbf{Y}, \mathbf{Y}^{[t]}$ be random variables on $\{0,1\}^d$ such that $\mathbf{A}$ is independent of $(\mathbf{B}, \mathbf{Y}, \mathbf{Y}^{[t]})$. Moreover, $\mathrm{H}_\infty(\mathbf{A}) \geq k$ and $\mathbf{Y} = \mathbf{U}_d$.*
- *$\mathbf{X} = \mathbf{A} + \mathbf{B}$*
- *$\alpha, \alpha^1, \ldots, \alpha^t$ be a-bit strings s.t. $\alpha \neq \alpha^i$ for every $i \in [t]$.*

then

$$(\mathrm{ACB}(\mathbf{X}, \mathbf{Y}, \alpha) \approx_\varepsilon \mathbf{U}_m) \mid (\{\mathrm{ACB}(\mathbf{X}, \mathbf{Y}^i, \alpha^i)\}_{i\in[t]}).$$

In addition, we say ACB is strong if

$$(\mathrm{ACB}(\mathbf{X}, \mathbf{Y}, \alpha) \approx_\varepsilon \mathbf{U}_m) \mid (\mathbf{Y}, \mathbf{Y}^{[t]}, \{\mathrm{ACB}(\mathbf{X}, \mathbf{Y}^i, \alpha^i)\}_{i\in[t]}).$$

Note that an affine correlation breakers is more general than a standard one Definition 3.4 because one can take $\mathbf{B} = 0$ to get a standard correlation breaker.

To construct an affine correlation breaker, Li [Li16] adapted the construction of a (standard) correlation breaker in [CGL20] by replacing every strong seeded extractor in the construction with a *linear* one (e.g. Trevisan's extractor [Tre01]). It

seems plausible that one can get an affine correlation breaker with better parameters by applying the same strategy on more recent constructions of correlation breakers [CL16a, Coh16b, Coh17, Li17, Li19], and the entropy requirement for affine extractors should be improved correspondingly. However, there is an issue that prevents us from getting affine extractors with entropy requirement as good as of two-source extractors: the best known parameters of *linear* strong seeded extractors are not as good as their non-linear counterparts.

In fact, the lack of optimal linear seeded extractors affects the entropy requirement of affine extractors in two different steps.

- In the first step we take a strong linear seeded extractor $\mathrm{LExt} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ and enumerate $\mathrm{LExt}(\mathbf{X}, 1), \ldots, \mathrm{LExt}(\mathbf{X}, D)$ for all $D = 2^d$ possible seeds. To make sure that this step can be completed in polynomial time, we need $d = O(\log(n))$. In addition, the error of LExt needs to be $1/\operatorname{poly}(n)$ for the BDT error reduction (see Section 3.5) to work. Furthermore, we also need the output length m to be long enough for the affine correlation breaker applied in the later step, and all known correlation breakers before our work ([CGL21]) require $m = \omega(\log(n)) > d$.[7]
- All known constructions of correlation breakers involve the alternating extraction process (Lemma 3.5), and this requires a strong seeded extractor with output length no shorter than its seed length.

In both steps we need a strong linear seeded extractor with output length no shorter than its seed length, but prior constructions of extractors with such guarantee have entropy requirement at least $\log^C(n)$ for some $C \geq 2$. To get an affine extractor for almost logarithmic entropy, we bypass this problem in two different ways.

[7] A very recent breakthrough by Li [Li23] showed that $m = O(\log(n))$ suffices.

- For the first step, we utilize the fact that the output length m is short ($\mathrm{polylog}(n)$) to construct a strong linear seeded extractor with seed length $O(\log(n))$. There is a small drawback that the error of our extractor is $n^{-1/c(n)}$ for some slow-growing function c. However, it turns out that $c(n)$ is small enough so that this is not a problem in the BDT error reduction framework.
- For the construction of correlation breakers, we revisit the argument by Li [Li16] and show that we don't need to replace every extractor with a linear one. To elaborate, recall that in the alternating extraction process Lemma 3.5 between two independent sources $\mathbf{X}$ and $\mathbf{Y}$, we need to apply two extractors $\mathrm{Ext}_x : \{0,1\}^n \times \{0,1\}^{d_y} \to \{0,1\}^{d_x}$ and $\mathrm{Ext}_y : \{0,1\}^m \times \{0,1\}^{d_x} \to \{0,1\}^{d_y}$ on $\mathbf{X}$ and $\mathbf{Y}$ respectively. We show that in the case that $\mathbf{X} = \mathbf{A} + \mathbf{B}$ where $\mathbf{A}$ is independent of $(\mathbf{B}, \mathbf{Y})$, it suffices to take only Ext_x to be linear. Therefore, because we can take $d_y > d_x$, the strong linear seeded extractor Ext_x can have seed length longer than the output length, which is easy to construct. (See Lemma 2.16.) As a bonus, we use a similar idea to show that any function applied on the $\mathbf{Y}$ side can be non-linear. This observation significantly simplifies our analysis: note that the construction we introduced in Section 3.4 is "affine-friendly" in the sense that every function applied on $\mathbf{X}$ is just a strong seeded extractor. Therefore we do not need to re-analyze any building block in the affine setting. In fact, we can even use this observation to get a black-box reduction from affine correlation breakers to standard correlation breakers with only a small loss. This shows that any improvement in the standard setting can be easily translated to the affine setting. In this chapter we introduce both the construction from scratch (first appeared in [CGL21]) and the construction based on reduction to standard correlation breaker (first appeared in [CL22]).

Finally we briefly discuss an assumption that we made in our discussion about

CZ framework in Chapter 3: we assumed that the output of a strong seeded extractor $\mathrm{Ext}(\mathbf{X}, i)$ for a good seed i is exactly uniform. This is of course not true in general, but it turns out that we can safely assume this if Ext is linear and $\mathbf{X}$ is affine, because of the following simple lemma from [Rao09].

Lemma 4.4 ([Rao09])**.** *Let* $\mathrm{LExt} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *be a* (k, ε)*-strong linear seeded extractor. Then for every* (n, k)*-affine source* $\mathbf{X}$*,*

$$\Pr_{s \sim \mathbf{U}_d}[\mathrm{LExt}(\mathbf{X}, s) \textit{ is uniform}] \geq 1 - 2\varepsilon.$$

Proof (sketch). Because $\mathrm{LExt}(\cdot, s)$ is linear, $\mathrm{LExt}(\mathbf{X}, s)$ is also uniform over an affine subspace. Therefore $\mathrm{LExt}(\mathbf{X}, s)$ is either exactly uniform or $1/2$-far from uniform. The bound follows by Markov's inequality. □

4.1.2 Our Results on Affine Correlation Breakers

In this section we summarize our results about affine correlation breakers. The first one is adapted from the construction we introduced in Chapter 3, which is an affine-friendly variant of the correlation breaker in [Li17]. For simplicity we assume that the output length is 1. This theorem is only used to showcase our main idea and will not be used in the proof of Theorem 4.1.

Theorem 4.5. *For every* $n, t, a, m \in \mathbb{N}$ *and* $\varepsilon > 0$*, there is an explicit* $(1, k, \varepsilon)$*-strong affine correlation breaker* $\mathrm{ACB} : \{0,1\}^n \times \{0,1\}^d \times \{0,1\}^a \to \{0,1\}$ *with* $k = O(t^2 \log(a) \log(\frac{na}{\varepsilon}))$ *and* $d = O(t^2 \log(a) \log(\frac{na}{\varepsilon}) + t^3 \log(na/\varepsilon))$*.*

Next we introduce our reduction to correlation breakers. For this reduction, we need a more generalized definition of correlation breakers that allows $\mathbf{X}$ to be tampered. This generalized definition is naturally satisfied by the constructions that

are based on alternating extraction, including the one in [Li19] that improves over [Li17]. In addition, we omit the independence number t because $t = 1$ suffices for our reduction.

Definition 4.6. $\mathrm{CB} : \{0,1\}^n \times \{0,1\}^d \times \{0,1\}^a \to \{0,1\}^m$ *is a correlation breaker for entropy* k *with error* ε *(or a* (k,ε)*-correlation breaker for short) if for every distributions* $\mathbf{X}, \mathbf{X}' \in \{0,1\}^n$*,* $\mathbf{Y}, \mathbf{Y}' \in \{0,1\}^d$ *and strings* $\alpha, \alpha' \in \{0,1\}^a$ *such that*

- $\mathrm{H}_\infty(\mathbf{X}) \geq k$ *and* $\mathbf{Y}$ *is uniform*
- $(\mathbf{X}, \mathbf{X}')$ *is independent of* $(\mathbf{Y}, \mathbf{Y}')$
- $\alpha \neq \alpha'$

it holds that

$$\mathrm{CB}(\mathbf{X}, \mathbf{Y}, \alpha) \approx_\varepsilon \mathbf{U}_m \mid \mathrm{CB}(\mathbf{X}', \mathbf{Y}', \alpha').$$

We say CB *is strong if*

$$\mathrm{CB}(\mathbf{X}, \mathbf{Y}, \alpha) \approx_\varepsilon \mathbf{U}_m \mid (\mathbf{Y}, \mathbf{Y}', \mathrm{CB}(\mathbf{X}', \mathbf{Y}', \alpha')) .$$

Theorem 4.7. *Let* C *be a large enough constant. Suppose that there exists an explicit* (d_0, ε)*-strong correlation breaker* $\mathrm{CB} : \{0,1\}^d \times \{0,1\}^{d_0} \times \{0,1\}^a \to \{0,1\}^{C \log^2(t+1) \log(n/\varepsilon)}$ *for some* $n, t \in \mathbb{N}$*. Then there exists an explicit strong* $(t, k, O(t\varepsilon))$*-strong affine correlation breaker* $\mathrm{ACB} : \{0,1\}^n \times \{0,1\}^d \times \{0,1\}^a \to \{0,1\}^m$ *where* $k = O(td_0 + tm + t^2 \log(n/\varepsilon))$ *and* $d = O(td_0 + m + t \log^3(t+1) \log(n/\varepsilon))$*.*

We can plug in the following correlation breaker from [Li19] to our reduction and get a stronger result than Theorem 4.5.

Theorem 4.8 ([Li19]). *There exists an explicit* (d, ε)*-strong correlation breaker* $\{0,1\}^n \times \{0,1\}^d \times \{0,1\}^a \to \{0,1\}^m$ *where* $d = O\left(m + \log(n/\varepsilon) \cdot \frac{\log(a)}{\log\log(a)}\right)$.

Theorem 4.9. *For every $m, a, t \in \mathbb{N}$ and $\varepsilon > 0$ there exists an explicit strong (t, k, ε)-affine correlation breaker* $\mathrm{ACB} : \{0,1\}^n \times \{0,1\}^d \times \{0,1\}^a \to \{0,1\}^m$ *where* $d = O\left(t \log\left(\frac{n}{\varepsilon}\right) \cdot \left(\frac{\log(a)}{\log\log(a)} + \log^3(t)\right)\right)$ *and* $k = O\left(tm + t\log\left(\frac{n}{\varepsilon}\right) \cdot \left(\frac{\log(a)}{\log\log(a)} + t\right)\right)$.

Subsequent works. In a recent breakthrough, Li [Li23] showed how to construct a correlation breaker for output length $O(\log(n/\varepsilon))$ with $d = k = O(\log(n/\varepsilon))$. By our reduction (Theorem 4.7) this also implies an affine correlation breaker with similar parameters. Based on the new affine correlation breaker, one can obtain an affine extractor for $O(\log(n))$ entropy, which is optimal up to a constant factor.

Organization. In Section 4.2 we construct the strong linear seeded extractor that will be used in the first step of our affine extractor. In Section 4.3 we prove Theorem 4.1 assuming that Theorem 4.7 is correct. In Section 4.4 we prove our results on affine correlation breakers (Theorem 4.5 and Theorem 4.7).

4.2 Linear Seeded Extractors with Short Output

In this section we show how to construct a strong linear seeded extractor that has $O(\log(n))$ seed length and slightly sub-polynomial error for short output, and apply BDT error reduction [BDT19] on such an extractor. More generally, we prove the following theorem.

Theorem 4.10. *For every $c = c(n), m = m(n)$ and every $\varepsilon = \varepsilon(n) > 0$, there exists a family of explicit (k, ε)-strong linear seeded extractor* $\mathrm{LExt} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{cm}$ *such that $k = O(cm + c\log c \log(cm/\varepsilon))$ and $d = O(m + \log n + \log^2(c)\log(cm/\varepsilon))$.*

By taking $m = \log n$ and $\varepsilon = n^{-1/\log^2(c)}$ we get strong linear seeded extractor with seed length $O(\log(n))$.

Corollary 4.11. *For every $c = c(n) < 2^{\sqrt[3]{\log n}}$, there exists an explicit (k, ε)-strong linear seeded extractor* $\mathrm{LExt} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{c \log n}$ *such that $k = O(c \log(n))$, $d = O(\log(n))$ and $\varepsilon = n^{-1/\log^2(c)}$.*

Our construction crucially rely on the fact that a block source (Definition 2.7) can be extracted/condensed with the same seed if the given extractor/condenser is strong. The construction works as follows. First we apply the GUV condenser (Lemma 2.14) to get a $((1+\gamma)k, k)$-source for some small constant $\gamma > 0$. This costs seed length roughly $O(\log(n))$. Then we repeat the following step: given a block source with t blocks (Definition 2.7), first we split each block into two halves and use Lemma 4.13 to argue that we get a block source with $2t$ blocks. Then we take a new seed with length $O(\log(k/\varepsilon))$ and use this seed to apply GUV condenser on all the blocks. We repeat this step for $\log(c)$ rounds until there are c blocks with entropy $O(m)$, and finally we use a fresh seed with length $O(m + \log(1/\varepsilon))$ to extract m bits from each block. Intuitively the seed length should be $O(m + \log(n) + \log(c)\log(m/\varepsilon))$, but we need an extra $\log(c)$ factor in the last term because we actually need to take $\gamma = O(1/\log(c))$. To formally prove the argument above, first we prove the following "block-source condensing" lemma.

Lemma 4.12 (block-source condensing)**.** *Let $\mathbf{X} = (\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_t)$ be a (t, n, k)-source and* $\mathrm{Con} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *be a (k, k', ε)-strong condenser. Let $\mathbf{Y} = \mathbf{U}_d$. Define*

$$\mathrm{BlockCon}(\mathbf{X}, \mathbf{Y}) := (\mathrm{Con}(\mathbf{X}_1, \mathbf{Y}), \mathrm{Con}(\mathbf{X}_2, \mathbf{Y}), \ldots, \mathrm{Con}(\mathbf{X}_t, \mathbf{Y})).$$

Then $(\mathbf{Y}, \mathrm{BlockCon}(\mathbf{X}, \mathbf{Y}))$ is $t\varepsilon$-close to a distribution $(\mathbf{Y}, \mathbf{Z}_1, \mathbf{Z}_2, \ldots, \mathbf{Z}_t)$ such that $(\mathbf{Z}_1, \ldots, \mathbf{Z}_t)$ is a (t, m, k')-block source conditioned on any fixing of $\mathbf{Y}$.

Proof. We prove by induction that for every $0 \le i \le t$, there exists $(\mathbf{Z}_{i+1}, \dots, \mathbf{Z}_t)$ such that

$$(\mathrm{Con}(\mathbf{X}_{i+1}, \mathbf{U}_d), \dots, \mathrm{Con}(\mathbf{X}_t, \mathbf{U}_d)) \approx_{(t-i)\varepsilon} (\mathbf{Z}_{i+1}, \dots, \mathbf{Z}_t) \mid (\mathbf{X}_{\le i}, \mathbf{Y}) \tag{4.1}$$

and $(\mathbf{Z}_{i+1}, \dots, \mathbf{Z}_t)$ is a $(t-i, m, k')$-block source conditioned on any fixing of $\mathbf{X}_{\le i}, \mathbf{Y}$. Note that the $i = 0$ case is what we want to prove in this lemma.

The base case $i = t$ is trivial. Now assume there exists $(\mathbf{Z}_{i+1}, \dots, \mathbf{Z}_t)$ which satisfies the induction hypothesis. We show that there exists $\mathbf{Z}_i$ such that

$$(\mathrm{Con}(\mathbf{X}_{i+1}, \mathbf{U}_d), \dots, \mathrm{Con}(\mathbf{X}_t, \mathbf{U}_d)) \approx_{(t-i)\varepsilon} (\mathbf{Z}_{i+1}, \dots, \mathbf{Z}_t) \mid (\mathbf{X}_{< i}, \mathbf{Y}) \tag{4.2}$$

and $(\mathbf{Z}_i, \mathbf{Z}_{i+1}, \dots, \mathbf{Z}_t)$ is a $(t-i+1, m, k')$-block source conditioned on any fixing of $\mathbf{X}_{<i}, \mathbf{Y}$. First note that $\mathbf{X}_i$ has min-entropy at least k conditioned on any fixing of $\mathbf{X}_{<i}$. By the definition of strong condenser and by there exists $\mathbf{Z}_i$ such that $(\mathrm{Con}(\mathbf{X}_i) \approx_\varepsilon \mathbf{Z}_i) \mid (\mathbf{X}_{<i}, \mathbf{Y})$. By Lemma 2.32 we can find a joint distribution $(\mathbf{X}_{<i}, \mathbf{Y}, \mathbf{Z}_i, \mathbf{Z}_{i+1}, \dots, \mathbf{Z}_t)$ such that

$$(\mathrm{Con}(\mathbf{X}_i) \approx_\varepsilon \mathbf{Z}_i) \mid (\mathbf{Z}_{i+1}, \dots, \mathbf{Z}_t, \mathbf{X}_{<i}, \mathbf{Y}), \tag{4.3}$$

$\mathrm{H}_\infty(\mathbf{Z}_i \mid \mathbf{X}_{<i}, \mathbf{Y}) \ge k'$ and $\mathbf{Z}_i \leftrightarrow (\mathbf{X}_{<i}, \mathbf{Y}) \leftrightarrow (\mathbf{Z}_{i+1}, \dots, \mathbf{Z}_t)$ is a Markov chain. Since $(\mathbf{Z}_{i+1}, \dots, \mathbf{Z}_t)$ is a $(t-i, m, k')$-block source conditioned on any fixing of $(\mathbf{X}_{<i}, \mathbf{Y})$, it's still a $(t-i, m, k')$-block source after further fixing $\mathbf{Z}_i$. Therefore $(\mathbf{Z}_i, \dots, \mathbf{Z}_t)$ is a $(t-i+1, m, k')$-block source conditioned on any fixing of $(\mathbf{X}_{<i}, \mathbf{Y})$. By adding $\mathrm{Con}(\mathbf{X}_i, \mathbf{Y})$ to (4.1), applying triangle inequality with (4.3) and then removing $\mathbf{X}_i$, we get (4.2). □

We also need the following lemma, which is a simple corollary of Lemma 2.8 by induction.

Lemma 4.13. *Let* $\mathbf{X} = (\mathbf{X}_1, \ldots, \mathbf{X}_t)$ *be a* $(t, 2n, 2n - \Delta)$*-source. Define* $\mathrm{Split}(\mathbf{X}) = (\mathbf{Y}_1, \ldots, \mathbf{Y}_{2t})$ *such that for every* $i \in [t]$, $\mathbf{X}_i = \mathbf{Y}_{2i-1} \circ \mathbf{Y}_{2i}$ *and* $\mathbf{Y}_{2i-1}, \mathbf{Y}_{2i}$ *have* n *bits each. Then* $\mathrm{Split}(\mathbf{X})$ *is* $t\varepsilon$*-close to a* $(2t, n, n - \Delta - \log(1/\varepsilon))$*-block source.*

Now we are ready to prove Theorem 4.10.

Proof. Let $h = \log(c), k_0 = k, \varepsilon_0 = \frac{\varepsilon}{4c}$ and $k_1, \ldots, k_h, m_0, \ldots, m_h, d_0, \ldots, d_h, d_{ext} \in \mathbb{N}$ be parameters to be defined later. Given a (n, k_0)-source $\mathbf{X}$, first we take some uniform seeds $(\mathbf{Y}_0, \mathbf{Y}_1, \ldots, \mathbf{Y}_h)$, each having length $d_0, d_1, \ldots, d_h$ respectively. Our first step is to iteratively construct $\mathbf{X}_0, \mathbf{X}_1, \ldots, \mathbf{X}_h$ such that for every $0 \le i \le h$,

$$(\mathbf{Y}_{\le i}, \mathbf{X}_i) \approx_{2^{i+1}\varepsilon_0} (\mathbf{Y}_{\le i}, \mathbf{Z}_i) \tag{4.4}$$

where $\mathbf{Z}_i$ is a $(2^i, m_i, k_i)$-block source conditioned on any fixing of $\mathbf{Y}_{\le i}$. To do this, first we define

$$\mathbf{X}_0 := \mathrm{LCon}_0(\mathbf{X}, \mathbf{Y}_0),$$

where $\mathrm{LCon}_0 : \{0,1\}^n \times \{0,1\}^{d_0} \to \{0,1\}^{m_0}$ is a (k_0, ε_0)-strong linear lossless condenser. Then the $i = 0$ case of (4.4) follows by definition of strong condenser. Next, for every $i \in [h]$, we define

$$\mathbf{X}_i := \mathrm{Split}(\mathrm{BlockCon}_i(\mathbf{X}_{i-1}, \mathbf{Y}_i)),$$

where Split is from Lemma 4.13 and $\mathrm{BlockCon}_i$ is the block-source condenser from Lemma 4.12 instantiated with a (k_i, ε_0)-strong linear lossless condenser $\mathrm{LCon}_i : \{0,1\}^{m_{i-1}} \times \{0,1\}^{d_i} \to \{0,1\}^{2m_i}$. Observe that conditioned on any fixing of $\mathbf{Y}_{<i}$, by Lemma 4.12 there exists $\mathbf{Z}'_i$ such that

$$(\mathbf{Z}'_i, \mathbf{Y}_i) \approx_{(2^{i-1})\varepsilon_0} (\mathrm{BlockCon}_i(\mathbf{Z}_{i-1}, \mathbf{Y}_i), \mathbf{Y}_i)$$

and $\mathbf{Z}'_i$ is a $(2^{i-1}, 2m_i, k_{i-1})$-block source. By Lemma 4.13, if $k_{i-1} \ge m_i + k_i + \log(1/\varepsilon_0)$, then conditioned on any fixing of $\mathbf{Y}_{\le i}$ there exists a $(2^i, m_i, k_i)$-block source $\mathbf{Z}_i$ such

that

$$\begin{aligned}(\mathbf{Y}_{<i}, \mathbf{Y}_i, \mathbf{X}_i) &\approx_{2^i\varepsilon_0} (\mathbf{Y}_{<i}, \mathbf{Y}_i, \mathrm{Split}(\mathrm{BlockCon}_i(\mathbf{Z}_{i-1}, \mathbf{Y}_i)))\\ &\approx_{2^{i-1}\varepsilon_0} (\mathbf{Y}_{<i}, \mathbf{Y}_i, \mathrm{Split}_i(\mathbf{Z}'_i))\\ &\approx_{2^{i-1}\varepsilon_0} (\mathbf{Y}_{<i}, \mathbf{Y}_i, \mathbf{Z}_i).\end{aligned}$$

Then (4.4) holds by triangle inequality. In the last step, we take one more uniform seed $\mathbf{Y}_{ext} \in \{0,1\}^{d_{ext}}$, and output

$$\mathrm{LExt}(\mathbf{X}, (\mathbf{Y}_{\le h}, \mathbf{Y}_{ext})) := \mathrm{BlockExt}(\mathbf{X}_h, \mathbf{Y}_{ext})$$

where $\mathrm{BlockExt}$ is the block-source condenser from Lemma 4.12 instantiated with a (k_h, ε_0)-strong linear seeded extractor $\mathrm{LHL} : \{0,1\}^{m_h} \times \{0,1\}^{d_{ext}} \to \{0,1\}^{\log n}$ (note that an extractor is a special case of condenser). By (4.4) and Lemma 4.12 we can conclude that LExt is a (k, ε)-strong linear extractor with seed length $d = \sum_{i=0}^{h} d_i + d_{ext}$. Moreover, observe that LExt is linear since all the operations in this construction are linear if $\mathbf{Y}_0, \ldots, \mathbf{Y}_h, \mathbf{Y}_{ext}$ are fixed.

Finally we need to set the undefined parameters and show that there exist explicit constructions of $\mathrm{LCon}_0, \ldots, \mathrm{LCon}_h$ and LHL. First we set $k_h = m + 2\log(1/\varepsilon_0)$, so that LHL exists by Lemma 2.15. Next, for every $1 \le i \le h$, let LCon_i be the lossless condenser from Lemma 2.14 by setting $\alpha = \frac{1}{h+1}$. Then $2m_i = (1 + \frac{1}{h+1})k_{i-1} + d_i$ and $d_i = (h+2)\log(m_{i-1}k_{i-1}/\varepsilon_0) + O(1)$. If suffices to take $k_{i-1} = m_i + k_i + \log(1/\varepsilon_0)$, which can be rewritten as

$$k_{i-1} = 2\left(1 + \frac{1}{h}\right)\left(k_i + \frac{d_i}{2} + \log(1/\varepsilon_0)\right).$$

Set $k_{h-1}, \ldots, k_0$ based on this recurrence, and use the fact that $d_1 \ge \cdots \ge d_h$, we get that for every $0 \le i < h$,

$$k_i \le \left(2\left(1 + \frac{1}{h}\right)\right)^{h-i}(m + d_1 + 4\log(1/\varepsilon_0)),$$

and specifically $k_0 = O(c(m + d_1 + \log(1/\varepsilon_0)))$. Observe that it suffices to take $d_1 = O(h \log(cm_0k_0/\varepsilon)) = O(\log(c) \log(cm/\varepsilon))$ and $k_0 = O(cm + c \log(c) \log(cm/\varepsilon))$. Finally, for LCon_0, we take the condenser from Lemma 2.14 by setting $\alpha = 1$. Then we get $d_0 = O(\log(n/\varepsilon))$ and $m_0 = 2k_0 + d_0$. The total seed length of this extractor is

$$d = \sum_{i=0}^{h} d_i + d_{ext} \leq d_0 + hd_1 + m_h = O(m + \log^2(c) \log(cm/\varepsilon)).$$

□

Finally we apply BDT error reduction on Corollary 4.11 to get the following "somewhere random extractor" LSREXT, i.e., for most of the seeds s we have that $\mathrm{LSREXT}(\mathbf{X}, s, z)$ is uniform for some z. Note that we only consider affine sources $\mathbf{X}$ so that we can apply Lemma 4.4 to argue that most outputs are *exactly* uniform.

Theorem 4.14. *For every constant $\gamma > 0$, there exists a constant C that satisfies the following. For every $n \in \mathbb{N}$ and every $c = c(n) < 2^{\sqrt[3]{\log n}}$, there is an explicit function $\mathrm{LSREXT} : \{0,1\}^n \times [D] \times [A] \to \{0,1\}^{c \log n}$ such that*

- $D \leq n^C$
- $A \leq C \log^2(c)$
- *For every fixed $s \in [D], z \in [A]$, the function $\mathrm{LSREXT}_{s,z}(x) := \mathrm{LSREXT}(x, s, z)$ is linear.*
- *There is some $k = O(c \log(n))$ such that for every (n, k)-affine source $\mathbf{X}$, there exists a subset $B \subseteq [D]$ of size at most D^γ such that for every $s \in [D] \backslash B$, $\exists z \in [A]$ s.t. $\mathrm{LSREXT}(\mathbf{X}, s, z)$ is uniform.*

Proof. Let $\Gamma : [D] \times [A] \to [D' = 2^{d'}]$ be the $(D^\gamma, 3\varepsilon)$-disperser from Lemma 2.21. Note that $D = 2^{2d'/\gamma} = 2^{O(d')}$ and $A = O(\frac{\log D}{\log(1/\varepsilon)}) = O(\frac{d'}{\log(1/\varepsilon)})$. Define

$$\mathrm{LSREXT}(x, s, z) := \mathrm{LEXT}(x, \Gamma(s, z)),$$

where LExt is the strong seeded extractor from Corollary 4.11. Observe that $\mathrm{LSRExt}_{s,z}$ is linear for every s, z since LExt is linear. It remains to prove the last property. Let B be the set which consists of every s s.t. $\forall z \in [A]$, $\mathrm{LExt}(\mathbf{X}, \Gamma(s,z))$ is not uniform. By Lemma 4.4, the number of seed $y \in [D']$ such that $\mathrm{LExt}(\mathbf{X}, \Gamma(s,z))$ is not uniform is at most $2\varepsilon D'$. Therefore $|\Gamma(B)| \leq 2\varepsilon D' < 3\varepsilon D'$, which implies that $|B| < D^{\gamma}$. □

4.3 Construction of Affine Extractors

In this section we formally prove Theorem 4.1, assuming that Theorem 4.9 is true. We leave the proof of Theorem 4.9 and other results about affine correlation breakers to the next section.

Theorem 4.15 (Theorem 4.1, restated)**.** *For every constant $\varepsilon > 0$, there exists a constant C such that for every large enough n, there exists an explicit extractor $\mathrm{AffExt} : \{0,1\}^n \to \{0,1\}$ for (n,k)-affine source with error ε for any $k \geq C \log(n) \log\log(n) \log\log\log^3(n)$.*

Proof. Let $\mathbf{X}$ be a (n,k)-affine source. Let $t = O(\frac{\log^2(1/\varepsilon)}{\varepsilon^2})$ be large enough so that the majority function Maj is an explicit extractor with error $\varepsilon/2$ for $(D^{0.4}, t)$-NOBF sources with D bits by Lemma 3.2. Let $\mathrm{LSRExt} : \{0,1\}^n \times [D] \times [A] \to \{0,1\}^{c \log n}$ be a strong linear seeded somewhere random extractor from Theorem 4.14 where $\delta = 0.4$, and assume that c and k is large enough so that

- k satisfies the entropy requirement in Theorem 4.14.
- An explicit (tA, k_0, a, γ)-affine correlation breaker $\mathrm{ACB} : \{0,1\}^n \times \{0,1\}^{c\log(n)} \times \{0,1\}^a \to \{0,1\}$ exists based on Theorem 4.9, where $k_0 = k - tA(c\log(n))$, $a = \log(AD)$ and $\gamma = \varepsilon/(2tD^t)$.

We set the parameters later. Now for every $s \in [D], z \in [A]$, compute

$$\mathbf{Y}_{s,z} := \mathrm{LSRExt}(\mathbf{X}, s, z)$$

$$\mathbf{R}_{s,z} := \mathrm{ACB}(\mathbf{X}, \mathbf{Y}_{s,z}, (s, z))$$

$$\mathbf{P}_s := \oplus_{j=1}^{A} \mathbf{R}_{s,j}.$$

Then the output is

$$\mathrm{AffExt}(\mathbf{X}) := \mathrm{Maj}(\mathbf{P}_1, \ldots, \mathbf{P}_D).$$

Observe that the running time is polynomial, since all the functions are explicit and $D, A \leq n^{O(1)}$. Next we prove the correctness of this construction. First note that by Theorem 4.14 there exists a set $B \in [D]$ of size at most $D^{0.4}$ such that for $\forall s \in [D] \backslash B, \exists z \in [A]$ s.t. $\mathbf{Y}_{s,z}$ is uniform. Now consider any $T \subseteq [D] \backslash B$ such that $|T| = t$. Let $T = \{s_1, s_2, \ldots, s_t\}$. For every $i \in [t]$, let $z_i \in [A]$ be the index such that $\mathbf{Y}_{s_i,z_i}$ is uniform. Observe that for every $i \in [t], j \in [A]$, the function $\mathrm{LSRExt}(\cdot, s_i, j)$ is linear, and so is their concatenation. Since their total length is $tA(c\log(n))$, by Lemma 4.2 there exists $\mathbf{A}, \mathbf{B}$ such that $\mathbf{A} + \mathbf{B} = \mathbf{X}$, $\mathbf{A}$ is independent of $(\mathbf{B}, \{\mathbf{Y}_{s_i,j}\}_{i \in [t], j \in [A]})$, and $\mathrm{H}_\infty(\mathbf{A}) \geq k - tA(c\log(n) = k_0$. Then for every $i \in [t]$, by the definition of ACB we get that for every $i \in [t]$,

$$(\mathbf{R}_{s_i,z_i} \approx_\gamma \mathbf{U}_1) \mid \left(\{\mathbf{R}_{s_i,j}\}_{j \in [A] \backslash \{z_i\}}, \{\mathbf{R}_{s',j}\}_{s' \in T \backslash \{s_i\}, j \in [A]}\right).$$

This implies that

$$(\mathbf{P}_{s_i} \approx_\gamma \mathbf{U}_1) \mid \left(\{\mathbf{P}_{s'}\}_{s' \in T \backslash \{s_i\}}\right),$$

and then

$$(\mathbf{P}_{s_1}, \ldots, \mathbf{P}_{s_{i-1}}, \mathbf{P}_{s_i}, \mathbf{U}_{t-i}) \approx_\gamma (\mathbf{P}_{s_1}, \ldots, \mathbf{P}_{s_{i-1}}, \mathbf{U}_1, \mathbf{U}_{t-i}).$$

By triangle inequality we eventually get

$$(\mathbf{P}_{s_1}, \mathbf{P}_{s_2}, \ldots, \mathbf{P}_{s_t}) \approx_{t\gamma} \mathbf{U}_t.$$

In other words, for every $T \subseteq [D] \backslash B$ s.t. $|T| = t$, we have $\mathbf{P}_T$ is $t\gamma$-close to $\mathbf{U}_t$. By Lemma 2.4 we get that $(\mathbf{P}_1, \ldots, \mathbf{P}_D)$ is $\varepsilon/2$-close to a $(D^{0.4}, t)$-NOBF source. Therefore we can conclude that $\mathrm{AffExt}(\mathbf{X}) \approx_\varepsilon \mathbf{U}_1$ because Maj is an extractor for $(D^{0.4}, t)$-NOBF source with error $\varepsilon/2$.

Finally we consider the parameter restriction of c, k. First note that assuming $c < (\log\log(n))^{10}$ we have $t = O(1)$, $A = O(\log^2(c))$, $\gamma = n^{-O(1)}$ and $a = O(\log(n))$, so

$$\begin{aligned} c\log(n) &= O\left(tA\log\left(\frac{n}{\gamma}\right) \cdot \left(\frac{\log(a)}{\log\log(a)} + \log^3(tA)\right)\right) \\ &= O(\log^2(c)\log(n)\log\log(n)/\log\log\log(n)). \end{aligned}$$

It suffices to take $c = O(\log\log(n)\log\log\log(n))$. In addition, note that it suffices to take

$$\begin{aligned} k_0 &= O\left(tAc\log(n) + tA\log\left(\frac{n}{\gamma}\right) \cdot \left(\frac{\log(a)}{\log\log(a)} + tA\right)\right) \\ &= O(Ac\log(n) + A^2\log(n)) \\ &= O(\log(n)\log\log(n)\log\log\log^3(n)). \end{aligned}$$

Therefore $k = k_0 + tAc\log(n) = O(\log(n)\log\log(n)\log\log\log^3(n))$. □

4.4 Affine Correlation Breakers

In this section we prove our results on affine correlation breakers. First we briefly review the reason why strong linear seeded extractors work well in the setting of affine correlation breakers. Consider the alternating extraction process between $\mathbf{X} = \mathbf{A} + \mathbf{B}$ and $\mathbf{Y}$, where $\mathbf{A}$ is independent of $(\mathbf{B}, \mathbf{Y})$. Recall that the first step is to take a prefix $\mathbf{S}$ of $\mathbf{Y}$ and use it to extract from $\mathbf{X}$. For this step, although $\mathbf{S}$ is not independent of $\mathbf{X}$, we can use an argument by Rao [Rao09] that works as follows. If

we take a strong linear seeded extractor LExt to extract from $\mathbf{X}$, then we can write $\mathrm{LExt}(\mathbf{X}, \mathbf{S}) = \mathrm{LExt}(\mathbf{A}, \mathbf{S}) + \mathrm{LExt}(\mathbf{B}, \mathbf{S})$. Because $\mathbf{A}$ is independent of $\mathbf{S}$ and LExt, we know that $\mathrm{LExt}(\mathbf{A}, \mathbf{S})$ is uniform even after conditioned on $\mathbf{S}$. In addition, after conditioned on $\mathbf{S}$, $\mathrm{LExt}(\mathbf{A}, \mathbf{S})$ becomes independent of $\mathrm{LExt}(\mathbf{B}, \mathbf{S})$, so $\mathrm{LExt}(\mathbf{A}, \mathbf{S})$ is still uniform. Now let $\mathbf{R} = \mathrm{LExt}(\mathbf{X}, \mathbf{S}), \mathbf{R}_A = \mathrm{LExt}(\mathbf{A}, \mathbf{S}), \mathbf{R}_B = \mathrm{LExt}(\mathbf{B}, \mathbf{S})$, and recall that our next step is to use $\mathbf{R}$ to extract from $\mathbf{Y}$. Again $\mathbf{R}$ is not independent of $\mathbf{Y}$. However, if we condition on $\mathbf{R}_B$, then $\mathbf{R} = \mathbf{R}_A + \mathbf{R}_B$ becomes independent of $\mathbf{Y}$ because $\mathbf{R}_A$ is independent of $\mathbf{Y}$, and one can argue that $\mathbf{Y}$ still have enough entropy by chain rule (Lemma 2.10). Therefore we can still use $\mathbf{R}$ to extract from $\mathbf{Y}$. Our main observation is that we do not need a linear seeded extractor in this step. More generally, for any function that takes a high-entropy source $\mathbf{Y}$ and an independent seed $\mathbf{R}$ to complete some task, we can apply the same argument. Based on this observation, first we show how to construct an affine correlation breaker based on the construction of standard correlation breakers in Section 3.4.

Proof of Theorem 4.5. Let $\varepsilon_0 = (\varepsilon/Ca)$ for some large enough constant C. Consider $d_x = O(t \log(n/\varepsilon_0)), d_y = O(t \log(n/\varepsilon_0))$ so that there are explicit constructions of the following building blocks. (v, d are chosen later.)

- $\mathrm{LExt} : \{0,1\}^n \times \{0,1\}^{d_y} \to \{0,1\}^{d_x}$, which is a $(d_x + 2\log(1/\varepsilon_0), \varepsilon_0)$-strong linear seeded extractor from Lemma 2.16.
- $\mathrm{laExt}_2 : \{0,1\}^d \times \{0,1\}^{d_x} \to (\{0,1\}^v)^2$ from Lemma 3.6 with error parameter ε_0.
- $\mathrm{IPM}_2 : (\{0,1\}^v)^2 \times \{0,1\}^d \times \{0,1\}^{d_x} \to (\{0,1\}^v)$ from Lemma 3.8 with error parameter ε_0.

Given $\mathbf{X} = \mathbf{A} + \mathbf{B}, \mathbf{Y}$ and advice $\alpha = (\alpha_1, \ldots, \alpha_a)$, $\mathrm{ACB}(\mathbf{X}, \mathbf{Y}, \alpha)$ is computed as follows. (Without loss of generality, assume that a is a power of 2. In addition, the

output length is d_x, which clearly implies Theorem 4.5.)

1. Take $\mathbf{W}_0$ to be a prefix of $\mathbf{Y}$ of length d_y.
2. Compute $\mathbf{S}_0 := \mathrm{LExt}(\mathbf{X}, \mathbf{W}_0)$.
3. Compute $(\mathbf{R}_0, \mathbf{R}_1) := \mathrm{laExt}_2(\mathbf{Y}, \mathbf{S}_0)$.
4. Define $(\mathbf{V}_{0,1}, \mathbf{V}_{0,2}, \ldots, \mathbf{V}_{0,(2a-1)}, \mathbf{V}_{0,2a}) := (\mathbf{R}_{\alpha_1}, \ldots, \mathbf{R}_{\alpha_a}, \mathbf{R}_{1-\alpha_1}, \ldots, \mathbf{R}_{1-\alpha_a})$.
5. For i from 1 to $h = \log(2a)$, repeat the following steps:
   (a) Take $\mathbf{W}_i$ to be a prefix of $\mathbf{V}_{i-1,1}$ of length d_y.
   (b) Compute $\mathbf{S}_i := \mathrm{LExt}(\mathbf{X}, \mathbf{W}_i)$.
   (c) For every $j \in [2a/2^i]$, compute $\mathbf{V}_{i,j} := \mathrm{IPM}_2((\mathbf{V}_{(i-1),(2j-1)}, \mathbf{V}_{(i-1),2j}), \mathbf{Y}, \mathbf{S}_i)$.
6. Output $\mathrm{ACB}(\mathbf{X}, \mathbf{Y}, \alpha) := \mathrm{LExt}(\mathbf{X}, \mathbf{V}_{h,1})$.

To prove that this construction works, define $\mathbf{S}_{A,i} := \mathrm{LExt}(\mathbf{A}, \mathbf{W}_i), \mathbf{S}_{B,i} := \mathrm{LExt}(\mathbf{B}, \mathbf{W}_i)$. We also use $\mathbf{V}_i$ to denote $(\mathbf{V}_{i,1}, \ldots, \mathbf{V}_{i,(2a/2^i)})$. First note that $(\mathbf{S}_{i,0} \approx_{\varepsilon_0} \mathbf{U}_{d_x}) \mid (\mathbf{W}_0, \mathbf{W}_0^{[t]}, \mathbf{B}_0, \mathbf{B}_0^{[t]})$. Then we define $2a$ sets $S_{0,1}, S_{0,2}, \ldots, S_{0,2a} \subseteq [t]$ as follows. Note that for every $i \in [t]$ s.t. $\alpha^i \neq \alpha$, there exists $j \in [a]$ s.t. $\alpha^i_j \neq \alpha_j$. If $\alpha_j = 1$ then $\mathbf{V}_{0,j} = \mathbf{R}_1$ and $\mathbf{V}^i_{0,j} = \mathbf{R}_0$, and in this case we add i into $S_{0,j}$. Otherwise $\mathbf{V}_{0,a+j} = \mathbf{R}_1$ and $\mathbf{V}^i_{0,a+j} = \mathbf{R}_0$, and in this case we add i into $S_{0,a+j}$. Observe that $\bigcup_{j=1}^{2a} S_{0,j} = [t]$, and for every $j \in [2a]$ we have

$$(\mathbf{V}_{0,j} \approx_{O(\varepsilon_0)} \mathbf{U}_v) \mid (\mathbf{W}_0, \mathbf{W}_0^{[t]}, \mathbf{S}_0, \mathbf{S}_0^{[t]}, \mathbf{B}_0, \mathbf{B}_0^{[t]}, \mathbf{V}_{0,j}^{S_{0,j}}),$$

because $\mathbf{V}_{0,j} = \mathbf{R}_0$ only when $S_{0,j}$ is empty, and if $S_{0,j}$ is non-empty then $\mathbf{V}^i_{0,j} = \mathbf{R}^i_0$ for every $i \in S_{0,j}$. Now for every $i \in [h]$ and $j \in [2a/2^i]$, define $S_{i,j} = S_{i-1,2j-1} \cup S_{i-1,2j}$. Note that $S_{h,1} = [t]$. Now for i from 0 to h, define $\mathbf{Z}_i = (\mathbf{W}_0, \mathbf{W}_0^{[t]}, \mathbf{S}_0, \mathbf{S}_0^{[t]}, \mathbf{B}_0, \mathbf{B}_0^{[t]})$. We inductively prove the following claims for i

from 1 to h. (We assume that the average conditional entropy of all random variables is large enough for all the lemmas we apply (which we will verify in the end).)

- $(\mathbf{A}, \mathbf{A}^{[t]}) \leftrightarrow \mathbf{Z}_{<i} \leftrightarrow (\mathbf{B}, \mathbf{B}^{[t]}, \mathbf{Y}, \mathbf{Y}^{[t]}, \mathbf{V}_{(i-1)}, \mathbf{V}^{[t]}_{(i-1)}, \mathbf{W}_i, \mathbf{W}_i^{[t]}, \mathbf{S}_{B,i}, \mathbf{S}^{[t]}_{B,i})$ forms a Markov chain.

  This is by induction hypothesis and by the fact that $\mathbf{W}_i, \mathbf{S}_{B,i}$ is a deterministic function of $\mathbf{B}, \mathbf{V}_{(i-1),1}$.

- $(\mathbf{W}_i \approx_{O(2^i\varepsilon_0)} \mathbf{U}_{d_y}) \mid \mathbf{Z}_{<i}$. (by (4.5) and Lemma 2.34)

- $\widetilde{\mathrm{H}}_\infty(\mathbf{A} \mid \mathbf{Z}_{<i}) \geq \mathrm{H}_\infty(\mathbf{A}) - i(t+1)d_x$ (by induction hypothesis and chain rule)

- $(\mathbf{S}_{A,i} \approx_{O(2^i\varepsilon_0)} \mathbf{U}_{d_x}) \mid (\mathbf{Z}_{<i}, \mathbf{W}_i, \mathbf{W}_i^{[t]}, \mathbf{S}_{B,i}, \mathbf{S}^{[t]}_{B,i})$ (by Lemma 2.34)

- $(\mathbf{A}, \mathbf{A}^{[t]}, \mathbf{S}_{A,i}, \mathbf{S}^{[t]}_{A,i}) \leftrightarrow (\mathbf{Z}_{<i}, \mathbf{W}_i, \mathbf{W}_i^{[t]}, \mathbf{S}_{B,i}, \mathbf{S}^{[t]}_{B,i}) \leftrightarrow (\mathbf{B}, \mathbf{B}^{[t]}, \mathbf{Y}, \mathbf{Y}^{[t]}, \mathbf{V}_{(i-1)}, \mathbf{V}^{[t]}_{(i-1)})$ forms a Markov chain (after fixing $\mathbf{W}_i, \mathbf{W}_i^{[t]}$ and then $\mathbf{S}_{B,i}, \mathbf{S}^{[t]}_{B,i}$)

- $(\mathbf{S}_i \approx_{O(2^i\varepsilon_0)} \mathbf{U}_{d_x}) \mid (\mathbf{Z}_{<i}, \mathbf{W}_i, \mathbf{W}_i^{[t]}, \mathbf{S}_{B,i}, \mathbf{S}^{[t]}_{B,i})$ (sum of $\mathbf{U}$ and any fixed string is still $\mathbf{U}$)

- $(\mathbf{A}, \mathbf{A}^{[t]}, \mathbf{S}_i, \mathbf{S}_i^{[t]}) \leftrightarrow (\mathbf{Z}_{<i}, \mathbf{W}_i, \mathbf{W}_i^{[t]}, \mathbf{S}_{B,i}, \mathbf{S}^{[t]}_{B,i}) \leftrightarrow (\mathbf{B}, \mathbf{B}^{[t]}, \mathbf{Y}, \mathbf{Y}^{[t]}, \mathbf{V}_{(i-1)}, \mathbf{V}^{[t]}_{(i-1)})$ forms a Markov chain. (This is because $\mathbf{S}_i = \mathbf{S}_{A,i} + \mathbf{S}_{B,i}$.)

- For every $j \in [2a/2^{i-1}]$, there exists $\widetilde{\mathbf{V}}_{(i-1),j}$ s.t.

$$\widetilde{\mathrm{H}}_\infty\left(\widetilde{\mathbf{V}}_{(i-1),j} \mid \left(\mathbf{V}^{S_{(i-1),j}}_{(i-1),j}, \mathbf{Z}_{<i}, \mathbf{W}_i, \mathbf{W}_i^{[t]}, \mathbf{B}_i, \mathbf{B}_i^{[t]}\right)\right) \geq v - (t+1)(d_x + d_y)$$

  and

$$\left(\mathbf{V}_{(i-1),j} \approx_{O(2^i\varepsilon_0)} \widetilde{\mathbf{V}}_{(i-1),j}\right) \mid \left(\mathbf{V}^{S_{(i-1),j}}_{(i-1),j}, \mathbf{Z}_{<i}, \mathbf{W}_i, \mathbf{W}_i^{[t]}, \mathbf{B}_i, \mathbf{B}_i^{[t]}\right).$$

  This $\widetilde{\mathbf{V}}_{(i-1),j}$ comes from the $\mathbf{U}_v$ in (4.5), which has $\widetilde{\mathrm{H}}_\infty\left(\widetilde{\mathbf{V}}_{(i-1),j} \mid \left(\mathbf{V}^{S_{(i-1),j}}_{(i-1),j}, \mathbf{Z}_{<i}\right)\right) = v$. The conditional entropy bound is by chain rule (Lemma 2.10).

- $\widetilde{\mathrm{H}}_\infty\left(\mathbf{Y} \mid \left(\mathbf{Z}_{<i}, \mathbf{W}_i, \mathbf{W}_i^{[t]}, \mathbf{B}_i, \mathbf{B}_i^{[t]}\right)\right) \geq d - i(t+1)(s_x + s_y)$.

- For every $j \in [2a/2^i]$,

$$\left(\mathbf{V}_{i,j} \approx_{O(2^{i+1}\varepsilon_0)} \mathbf{U}_v\right) \mid \left(\mathbf{V}_{i,j}^{S_{i,j}}, \mathbf{Z}_{<i}, \mathbf{W}_i, \mathbf{W}_i^{[t]}, \mathbf{S}_{B,i}, \mathbf{S}_{B,i}^{[t]}, \mathbf{S}_i, \mathbf{S}_i^{[t]}\right),$$

  which can be rewritten as

$$\left(\mathbf{V}_{i,j} \approx_{O(2^{i+1}\varepsilon_0)} \mathbf{U}_v\right) \mid \left(\mathbf{V}_{i,j}^{S_{i,j}}, \mathbf{Z}_{\le i}\right). \tag{4.5}$$

  This is by applying IPM_2 (Lemma 3.8) on $(\widetilde{\mathbf{V}}_{(i-1),(2j-1)}, \widetilde{\mathbf{V}}_{(i-1),2j})$. The error blows up by a factor 2 and will dominate the additive $O(\varepsilon_0)$ error in previous steps.

- $(\mathbf{A}, \mathbf{A}^{[t]}) \leftrightarrow \mathbf{Z}_{\le i} \leftrightarrow (\mathbf{B}, \mathbf{B}^{[t]}, \mathbf{Y}, \mathbf{Y}^{[t]}, \mathbf{V}_i, \mathbf{V}_i^{[t]})$ forms a Markov chain. (by fixing $\mathbf{S}_i, \mathbf{S}_i^{[t]}$)

Observe that (4.5) for $i = h, j = 1$ and one more round of Lemma 3.7 implies the correctness of our theorem. The error is $O(2^h\varepsilon_0) = O(a\varepsilon_0)$. Finally we analyze the entropy requirement. For IPM_2 and laExt_2 to work properly, we need $d - h(t+1)(s_x + s_y) \ge O(tv + t\log(n/\varepsilon))$ and $v - (t+1)(d_x + d_y) \ge O(t\log(d/\varepsilon))$. It suffices to take $v = O(t^2\log(na/\varepsilon))$ and $d = O(t^3\log(na/\varepsilon)) + t^2\log(a)\log(na/\varepsilon)$. For LExt to work properly, we need $\mathrm{H}_\infty(\mathbf{A}) = O(htd_x) = O(t^2\log(a)\log(na/\varepsilon))$. □

Next we prove the reduction from affine correlation breakers to correlation breakers. The construction is as follows. First we take a prefix of $\mathbf{Y}$ as the seed to extract a string $\mathbf{Z}$ from $\mathbf{X}$. Next we apply a standard correlation breaker which treats $\mathbf{Y}$ as the source and $\mathbf{Z}$ as the seed. Note that because the seed $\mathbf{Z}$ is on the $\mathbf{X}$ side, the correlation breaker does not need to be linear. Finally, we use the output of the standard correlation breaker as the seed to extract from $\mathbf{X}$.

A drawback of this simple reduction is that the resulting affine correlation breaker has worse dependence on the number of tampering t. To solve this problem, we only apply this reduction when constructing a 1-affine correlation breaker

based on a 1-correlation breaker. To construct a t-affine correlation breaker, we show an efficient way to strengthen a 1-affine correlation breaker to a t-affine correlation breaker based on the independence-merging lemma Lemma 3.7. The idea is as follows. Observe that even in the t-tampering setting, a 1-affine correlation breaker can still guarantee that the output is uniform when conditioned on *every single* tampered output (note that this does not mean the output is uniform when conditioned on multiple tampered outputs *simultaneously*.) Therefore we can apply $\log(t)$ rounds of alternating extractions to merge the independence of the output *with itself*. The formal proof of Theorem 4.7 is as follows.

Theorem 4.16 (Theorem 4.7, restated)**.** *Let C be a large enough constant. Suppose that there exists an explicit (d_0, ε)-strong correlation breaker $\mathrm{CB} : \{0,1\}^d \times \{0,1\}^{d_0} \times \{0,1\}^a \to \{0,1\}^{C \log^2(t+1) \log(n/\varepsilon)}$ for some $n, t \in \mathbb{N}$. Then there exists an explicit strong $(t, k, O(t\varepsilon))$-affine correlation breaker $\mathrm{ACB} : \{0,1\}^n \times \{0,1\}^d \times \{0,1\}^a \to \{0,1\}^m$ where $k = O(td_0 + tm + t^2 \log(n/\varepsilon))$ and $d = O(td_0 + m + t \log^3(t+1) \log(n/\varepsilon))$.*

Proof. Consider any $\mathbf{A}, \mathbf{B} \in \{0,1\}^n, \mathbf{Y}, \mathbf{Y}^{[t]} \in \{0,1\}^d, \mathbf{Z} \in \{0,1\}^*$ such that

- $\mathbf{A} \leftrightarrow \mathbf{Z} \leftrightarrow (\mathbf{B}, \mathbf{Y}, \mathbf{Y}^{[t]})$ forms a Markov chain
- $\widetilde{\mathrm{H}}_\infty(\mathbf{A} \mid \mathbf{Z}) \geq k$
- $(\mathbf{Y}, \mathbf{Z}) = (\mathbf{U}, \mathbf{Z})$,

and any $\alpha, \alpha^{[t]} \in \{0,1\}^a$ such that $\alpha \neq \alpha^i$ for every $i \in [t]$. Let $\mathbf{X} = \mathbf{A} + \mathbf{B}$. Our goal is to construct an algorithm ACB and prove that

$$(\mathrm{ACB}(\mathbf{X}, \mathbf{Y}, \alpha) \approx_{O(t\varepsilon)} \mathbf{U}_m) \mid (\{\mathrm{ACB}(\mathbf{X}, \mathbf{Y}^i, \alpha^i)\}_{i \in [t]}, \mathbf{Y}, \mathbf{Y}^{[t]}). \tag{4.6}$$

To keep the notation clean, in this proof we use the convention that $\mathbf{R} \leftrightarrow \mathbf{Z} \leftrightarrow \mathbf{S}$ should always be treated as $(\mathbf{R}, \mathbf{A}) \leftrightarrow \mathbf{Z} \leftrightarrow (\mathbf{S}, \mathbf{B}, \mathbf{Y}, \mathbf{Y}^{[t]})$. The algorithm ACB con-

sists of two phases. In the first phase we output a random variable with r bits that is uniform conditioned on every single tampered version, for some large enough r. We will see that $r = O(t \log(n/\varepsilon))$ suffices. In the second phase we strengthen its independence. In addition, if $r < m$ then we add one more round of alternating extraction to increase the output length. For the first two phases we need the following explicit extractors as building blocks, each having error ε. For LExt_0 and $\mathrm{LExt}_{\mathsf{m}}$ we take the explicit construction from Lemma 2.16. For Ext we take the GUV extractor from Lemma 2.17. For LExt_{sfm} we take the extractor in Theorem 4.14 with $c = t$. Observe that it suffices to take $d_0' = O(d_0 + \log(n/\varepsilon))$, $d_x = O(\log(n/\varepsilon))$ and $d_y = O(\log^2(t+1)\log(n/\varepsilon))$.

- $\mathrm{LExt}_0 : \{0,1\}^n \times \{0,1\}^{d_0'} \to \{0,1\}^{d_0}$
- $\mathrm{LExt}_{\mathsf{r}} : \{0,1\}^n \times \{0,1\}^{d_y} \to \{0,1\}^r$
- $\mathrm{LExt}_{\mathsf{m}} : \{0,1\}^r \times \{0,1\}^{d_y} \to \{0,1\}^{d_x}$
- $\mathrm{Ext} : \{0,1\}^d \times \{0,1\}^{d_x} \to \{0,1\}^{d_y}$

The first phase of $\mathrm{ACB}(\mathbf{X}, \mathbf{Y}, \alpha)$ consists of the following steps.

1. Let $\mathbf{S}_1 := \mathrm{Prefix}(\mathbf{Y}, d_0')$.
2. Compute $\mathbf{R}_1 := \mathrm{LExt}_0(\mathbf{X}, \mathbf{S}_1)$.
3. Compute $\mathbf{S}_2 := \mathrm{CB}(\mathbf{Y}, \mathbf{R}_1, \alpha)$.
4. Output $\mathbf{R}_2 := \mathrm{LExt}_{\mathsf{r}}(\mathbf{X}, \mathbf{S}_2)$.

To prove the correctness, first we define some random variables as follows: $\mathbf{R}_{1,\mathbf{A}} := \mathrm{LExt}_1(\mathbf{A}, \mathbf{S}_1)$, $\mathbf{R}_{1,\mathbf{B}} := \mathrm{LExt}_1(\mathbf{B}, \mathbf{S}_1)$, $\mathbf{R}_{2,\mathbf{A}} := \mathrm{LExt}_2(\mathbf{A}, \mathbf{S}_2)$ and $\mathbf{R}_{2,\mathbf{B}} = \mathrm{LExt}_2(\mathbf{B}, \mathbf{S}_2)$,

and let $\mathbf{Z}_0 = (\mathbf{Z}, \mathbf{S}_1, \mathbf{S}_1^{[t]}, \mathbf{R}_{1,\mathbf{B}}, \mathbf{R}_{1,\mathbf{B}}^{[t]}, \mathbf{R}_1, \mathbf{R}_1^{[t]}, \mathbf{S}_2, \mathbf{S}_2^{[t]})$. We will prove that for every $i \in [t]$,

$$(\mathbf{R}_{2,\mathbf{A}} \approx_{5\varepsilon} \mathbf{U}) \mid (\mathbf{R}_{2,\mathbf{A}}^i, \mathbf{Z}_0, \mathbf{R}_{2,\mathbf{B}}, \mathbf{R}_{2,\mathbf{B}}^{[t]}), \tag{4.7}$$

and

$$(\mathbf{A}, \mathbf{R}_{2,\mathbf{A}}, \mathbf{R}_{2,\mathbf{A}}^{[t]}) \leftrightarrow \mathbf{Z}_0 \leftrightarrow (\mathbf{B}, \mathbf{R}_{2,\mathbf{B}}, \mathbf{R}_{2,\mathbf{B}}^{[t]}, \mathbf{Y}, \mathbf{Y}^{[t]}) \text{ forms a Markov chain.} \tag{4.8}$$

Note that this means if we output $\mathbf{R}_2$ we already get a 1-affine correlation breaker. To prove (4.7) and (4.8), first note that by definition of LExt_0, we get $(\mathbf{R}_{1,\mathbf{A}} \approx_\varepsilon \mathbf{U}_{d_0}) \mid (\mathbf{Z}, \mathbf{S}_1, \mathbf{S}_1^{[t]})$. Fix $(\mathbf{S}_1, \mathbf{S}_1^{[t]})$. Since $(\mathbf{R}_{1,\mathbf{A}}, \mathbf{R}_{1,\mathbf{A}}^{[t]})$ are deterministic functions of $(\mathbf{A}, \mathbf{S}_1, \mathbf{S}_1^{[t]})$, and $(\mathbf{R}_{1,\mathbf{B}}, \mathbf{R}_{1,\mathbf{B}}^{[t]})$ are deterministic functions of $(\mathbf{B}, \mathbf{S}_1, \mathbf{S}_1^{[t]})$, $(\mathbf{R}_{1,\mathbf{A}}, \mathbf{R}_{1,\mathbf{A}}^{[t]})$ are independent of $(\mathbf{R}_{1,\mathbf{B}}, \mathbf{R}_{1,\mathbf{B}}^{[t]})$. Fix $(\mathbf{R}_{1,\mathbf{B}}, \mathbf{R}_{1,\mathbf{B}}^{[t]})$. Then $\mathbf{R}_{1,\mathbf{A}}$ is still close to uniform. Because $\mathbf{R}_1 = \mathbf{R}_{1,\mathbf{A}} + \mathbf{R}_{1,\mathbf{B}}$, this implies

$$(\mathbf{R}_1 \approx_\varepsilon \mathbf{U}_{d_0}) \mid (\mathbf{Z}, \mathbf{S}_1, \mathbf{S}_1^{[t]}, \mathbf{R}_{1,\mathbf{B}}, \mathbf{R}_{1,\mathbf{B}}^{[t]}).$$

Moreover, $\widetilde{\mathrm{H}}_\infty(\mathbf{Y} \mid \mathbf{Z}, \mathbf{S}_1, \mathbf{S}_1^{[t]}, \mathbf{R}_{1,\mathbf{B}}, \mathbf{R}_{1,\mathbf{B}}^{[t]}) \geq d - O(t(d_0 + \log(n/\varepsilon))) \geq d_0 + \log(1/\varepsilon)$. Because

$$(\mathbf{R}_1, \mathbf{R}_1^{[t]}) \leftrightarrow (\mathbf{Z}, \mathbf{S}_1, \mathbf{S}_1^{[t]}, \mathbf{R}_{1,\mathbf{B}}, \mathbf{R}_{1,\mathbf{B}}^{[t]}) \leftrightarrow (\mathbf{Y}, \mathbf{Y}^{[t]})$$

is a Markov chain, and because CB is a strong correlation breaker, for every $i \in [t]$ we have

$$(\mathbf{S}_2 \approx_{3\varepsilon} \mathbf{U}_{d_y}) \mid (\mathbf{S}_2^i, \mathbf{Z}, \mathbf{S}_1, \mathbf{S}_1^{[t]}, \mathbf{R}_{1,\mathbf{B}}, \mathbf{R}_{1,\mathbf{B}}^{[t]}, \mathbf{R}_1, \mathbf{R}_1^i).$$

Note that after fixing $\mathbf{R}_1$, $\mathbf{S}_2$ becomes independent of $\mathbf{R}_1^{[t]}$. Therefore

$$(\mathbf{S}_2 \approx_{3\varepsilon} \mathbf{U}_{d_y}) \mid (\mathbf{S}_2^i, \mathbf{Z}, \mathbf{S}_1, \mathbf{S}_1^{[t]}, \mathbf{R}_{1,\mathbf{B}}, \mathbf{R}_{1,\mathbf{B}}^{[t]}, \mathbf{R}_1, \mathbf{R}_1^{[t]}).$$

Fix $\mathbf{R}_1, \mathbf{R}_1^{[t]}$. Because $\mathbf{A}$ are independent of $\mathbf{S}_2, \mathbf{S}_2^{[t]}$, by Lemma 3.7 we can conclude that

$$(\mathbf{R}_{2,\mathbf{A}} \approx_{5\varepsilon} \mathbf{U}_r) \mid (\mathbf{R}_{2,\mathbf{A}}^i, \mathbf{Z}, \mathbf{S}_1, \mathbf{S}_1^{[t]}, \mathbf{R}_{1,\mathbf{B}}, \mathbf{R}_{1,\mathbf{B}}^{[t]}, \mathbf{R}_1, \mathbf{R}_1^{[t]}, \mathbf{S}_2, \mathbf{S}_2^{[t]})$$

which is exactly

$$(\mathbf{R}_{2,\mathbf{A}} \approx_{5\varepsilon} \mathbf{U}_r) \mid (\mathbf{R}_{2,\mathbf{A}}^i, \mathbf{Z}_0). \tag{4.9}$$

Finally, fix $\mathbf{S}_2, \mathbf{S}_2^{[t]}$. Since $(\mathbf{R}_{2,\mathbf{A}}, \mathbf{R}_{2,\mathbf{A}}^{[t]})$ are independent of $(\mathbf{R}_{2,\mathbf{B}}, \mathbf{R}_{2,\mathbf{B}}^{[t]})$, we get (4.8). Then because $\mathbf{R}_2 = \mathbf{R}_{2,\mathbf{A}} + \mathbf{R}_{2,\mathbf{B}}$, by (4.8) and (4.9) we get (4.7).

Next we move to the second phase. Define $\mathbf{W}_{0,\mathbf{A}} := \mathbf{R}_{2,\mathbf{A}}, \mathbf{W}_{0,\mathbf{B}} := \mathbf{R}_{2,\mathbf{B}}, \mathbf{W}_0 := \mathbf{R}_2$ and $h = \lceil \log t \rceil$. Then repeat the following steps for i from 1 to h:

1. Let $\mathbf{W}_{\mathsf{p},i-1} := \mathrm{Prefix}(\mathbf{W}_{i-1}, d_x)$.
2. Compute $\mathbf{Q}_{\mathsf{m},i-1} := \mathrm{Ext}(\mathbf{Y}, \mathbf{W}_{\mathsf{p},i-1})$.
3. Compute $\mathbf{V}_i := \mathrm{LExt}_{\mathsf{m}}(\mathbf{W}_{i-1}, \mathbf{Q}_{\mathsf{m},i-1})$.
4. Compute $\mathbf{Q}_{\mathsf{r},i} := \mathrm{Ext}(\mathbf{Y}, \mathbf{V}_i)$.
5. Compute $\mathbf{W}_i := \mathrm{LExt}_{\mathsf{r}}(\mathbf{X}, \mathbf{Q}_{\mathsf{r},i})$.

Note that Step $1-3$ are the "independence merging" steps, which computes $\mathbf{V}_i$ that is independent of every 2^i tampered versions. Since the length of $\mathbf{V}_i$ is shorter than $\mathbf{W}_i$, we use Step $4-5$ to recover the length and get $\mathbf{W}_i$ s.t. $|\mathbf{W}_i| = r$. We claim that each of $\mathbf{W}_i, \mathbf{Q}_{\mathsf{m},i}, \mathbf{V}_i, \mathbf{Q}_{\mathsf{r},i}$ is independent of every $\min(2^i, t)$ tampered versions, and in particular $(\mathbf{W}_h, \mathbf{W}_h^{[t]}) \approx (\mathbf{U}_r, \mathbf{W}_h^{[t]})$.

Formally, for every i from 1 to h, let $\mathbf{W}_{\mathsf{p},i-1,\mathbf{A}} := \mathrm{Prefix}(\mathbf{W}_{i-1,\mathbf{A}}, d_x)$, $\mathbf{W}_{\mathsf{p},i-1,\mathbf{B}} := \mathrm{Prefix}(\mathbf{W}_{i-1,\mathbf{B}}, d_x)$, $\mathbf{V}_{i,\mathbf{A}} := \mathrm{LExt}_{\mathsf{m}}(\mathbf{W}_{i-1,\mathbf{A}}, \mathbf{Q}_{\mathsf{m},i-1})$, $\mathbf{V}_{i,\mathbf{B}} := \mathrm{LExt}_{\mathsf{m}}(\mathbf{W}_{i-1,\mathbf{B}}, \mathbf{Q}_{\mathsf{m},i-1})$, $\mathbf{W}_{i,\mathbf{A}} := \mathrm{LExt}_{\mathsf{r}}(\mathbf{A}, \mathbf{Q}_{\mathsf{r},i})$ and $\mathbf{W}_{i,\mathbf{B}} := \mathrm{LExt}_{\mathsf{r}}(\mathbf{B}, \mathbf{Q}_{\mathsf{r},i})$. Moreover, for every $i \in [h]$, let

$$\mathbf{Z}_i := \left(\mathbf{Z}_{i-1}, \mathbf{W}_{\mathsf{p},i-1,\mathbf{B}}, \mathbf{W}_{\mathsf{p},i-1,\mathbf{B}}^{[t]}, \mathbf{W}_{\mathsf{p},i-1}, \mathbf{W}_{\mathsf{p},i-1}^{[t]}, \mathbf{Q}_{\mathsf{m},i-1}, \mathbf{Q}_{\mathsf{m},i-1}^{[t]}, \mathbf{V}_{i,\mathbf{B}}, \mathbf{V}_{i,\mathbf{B}}^{[t]}, \mathbf{V}_i, \mathbf{V}_i^{[t]}, \mathbf{Q}_{\mathsf{r},i}, \mathbf{Q}_{\mathsf{r},i}^{[t]}\right).$$

We want to prove the following claims for every $i \in [h]$ by induction:

- For every $T \subseteq [t]$ s.t. $|T| = 2^i$,

$$(\mathbf{W}_{i,\mathbf{A}} \approx_{(13\cdot 2^i - 8)\varepsilon} \mathbf{U}_r) \mid (\mathbf{W}_{i,\mathbf{A}}^T, \mathbf{Z}_i). \tag{4.10}$$

- The following is a Markov chain:

$$(\mathbf{A}, \mathbf{W}_{i,\mathbf{A}}, \mathbf{W}_{i,\mathbf{A}}^{[t]}) \leftrightarrow \mathbf{Z}_i \leftrightarrow (\mathbf{B}, \mathbf{W}_{i,\mathbf{B}}, \mathbf{W}_{i,\mathbf{B}}^{[t]}, \mathbf{Y}, \mathbf{Y}^{[t]}). \tag{4.11}$$

Note that by (4.7) and (4.8), the conditions above hold for $i = 0$. Now assume by induction that (4.10) and (4.11) hold for $i-1$, and we want to prove (4.10) and (4.11) for i. First, observe that because $\mathbf{W}_{\mathsf{p},i-1} = \mathbf{W}_{\mathsf{p},i-1,\mathbf{A}} + \mathbf{W}_{\mathsf{p},i-1,\mathbf{B}}$, by (4.10) and (4.11) for every $T_1 \subseteq [t]$ of size 2^{i-1},

$$(\mathbf{W}_{\mathsf{p},i-1} \approx_{(13\cdot 2^{i-1} - 8)\varepsilon} \mathbf{U}_r) \mid (\mathbf{W}_{\mathsf{p},i-1}^{T_1}, \mathbf{Z}_{i-1}, \mathbf{W}_{\mathsf{p},i-1,\mathbf{B}}, \mathbf{W}_{\mathsf{p},i-1,\mathbf{B}}^{[t]}).$$

Fix $(\mathbf{W}_{\mathsf{p},i-1,\mathbf{B}}, \mathbf{W}_{\mathsf{p},i-1,\mathbf{B}}^{[t]})$. Note that

$$(\mathbf{W}_{\mathsf{p},i-1}, \mathbf{W}_{\mathsf{p},i-1}^{[t]}) \leftrightarrow (\mathbf{Z}_{i-1}, \mathbf{W}_{\mathsf{p},i-1,\mathbf{B}}, \mathbf{W}_{\mathsf{p},i-1,\mathbf{B}}^{[t]}) \leftrightarrow (\mathbf{Y}, \mathbf{Y}^{[t]})$$

is a Markov chain. By Lemma 3.7 (similarly we omit the entropy requirement for $\mathbf{Y}$ for now and will verify it in the end), for every $T_1 \subseteq [t]$ of size 2^{i-1},

$$(\mathbf{Q}_{\mathsf{m},i-1} \approx_{(13\cdot 2^{i-1} - 6)\varepsilon} \mathbf{U}_{d_y}) \mid (\mathbf{Q}_{\mathsf{m},i-1}^{T_1}, \mathbf{Z}_{i-1}, \mathbf{W}_{\mathsf{p},i-1,\mathbf{B}}, \mathbf{W}_{\mathsf{p},i-1,\mathbf{B}}^{[t]}, \mathbf{W}_{\mathsf{p},i-1}, \mathbf{W}_{\mathsf{p},i-1}^{[t]}).$$

Next, fix $(\mathbf{W}_{\mathsf{p},i-1}, \mathbf{W}_{\mathsf{p},i-1}^{[t]})$. Now consider any $T \subseteq [t]$ s.t. $|T| = \min(2^i, t)$, and any T_1, T_2 s.t. $|T_1| = |T_2| = 2^{i-1}$ and $T_1 \cup T_2 = T$. By (4.10) there exists $\mathbf{W}'_{i-1,\mathbf{A}} = \mathbf{U}_r$ s.t.

$$(\mathbf{W}_{i-1,\mathbf{A}} \approx_{(13\cdot 2^{i-1} - 8)\varepsilon} \mathbf{W}'_{i-1,\mathbf{A}}) \mid (\mathbf{W}_{i-1,\mathbf{A}}^{T_2}, \mathbf{Z}_{i-1}, \mathbf{W}_{\mathsf{p},i-1,\mathbf{B}}, \mathbf{W}_{\mathsf{p},i-1,\mathbf{B}}^{[t]}, \mathbf{W}_{\mathsf{p},i-1}, \mathbf{W}_{\mathsf{p},i-1}^{[t]})$$

and

$$\widetilde{\mathrm{H}}_\infty \left(\mathbf{W}'_{i-1,\mathbf{A}} \mid \mathbf{W}_{i-1,\mathbf{A}}^{T_2}, \mathbf{Z}_{i-1}, \mathbf{W}_{\mathsf{p},i-1,\mathbf{B}}, \mathbf{W}_{\mathsf{p},i-1,\mathbf{B}}^{[t]}, \mathbf{W}_{\mathsf{p},i-1}, \mathbf{W}_{\mathsf{p},i-1}^{[t]} \right) \geq r - (t+1)d_x.$$

Let $\mathbf{Z}'_{i-1} := \left(\mathbf{Z}_{i-1}, \mathbf{W}_{\mathsf{p},i-1,\mathbf{B}}, \mathbf{W}^{[t]}_{\mathsf{p},i-1,\mathbf{B}}, \mathbf{W}_{\mathsf{p},i-1}, \mathbf{W}^{[t]}_{\mathsf{p},i-1}, \mathbf{Q}_{\mathsf{m},i-1}, \mathbf{Q}^{[t]}_{\mathsf{m},i-1}\right)$. By Lemma 3.7,

$$(\mathbf{V}_{i,\mathbf{A}} \approx_{(13\cdot 2^i-14)\varepsilon} \mathbf{U}_{d_x}) \mid \left(\mathbf{V}^T_{i,\mathbf{A}}, \mathbf{Z}'_{i-1}\right).$$

Fix $(\mathbf{Q}_{\mathsf{m},i-1}, \mathbf{Q}^{[t]}_{\mathsf{m},i-1})$. Note that $\mathbf{Z}'_{i-1}$ consists of exactly the random variables we have fixed so far. Because $\mathbf{V}_i = \mathbf{V}_{i,\mathbf{A}} + \mathbf{V}_{i,\mathbf{B}}$ and $(\mathbf{V}_{i,\mathbf{A}}, \mathbf{V}^{[t]}_{i,\mathbf{A}}) \leftrightarrow \mathbf{Z}'_{i-1} \leftrightarrow (\mathbf{V}_{i,\mathbf{B}}, \mathbf{V}^{[t]}_{i,\mathbf{B}})$ forms a Markov chain,

$$(\mathbf{V}_i \approx_{(13\cdot 2^i-12)\varepsilon} \mathbf{U}_{d_x}) \mid \left(\mathbf{V}^T_i, \mathbf{Z}'_{i-1}, \mathbf{V}_{i,\mathbf{B}}, \mathbf{V}^{[t]}_{i,\mathbf{B}}\right).$$

Next we fix $(\mathbf{V}_{i,\mathbf{B}}, \mathbf{V}^{[t]}_{i,\mathbf{B}})$. Since $(\mathbf{V}_i, \mathbf{V}^{[t]}_i) \leftrightarrow (\mathbf{Z}'_{i-1}, \mathbf{V}_{i,\mathbf{B}}, \mathbf{V}^{[t]}_{i,\mathbf{B}}) \leftrightarrow (\mathbf{Y}, \mathbf{Y}^{[t]})$, again by Lemma 3.7,

$$(\mathbf{Q}_{\mathsf{r},i} \approx_{(13\cdot 2^i-10)\varepsilon} \mathbf{U}_{d_y}) \mid \left(\mathbf{Q}^T_{\mathsf{r},i}, \mathbf{Z}'_{i-1}, \mathbf{V}_{i,\mathbf{B}}, \mathbf{V}^{[t]}_{i,\mathbf{B}}, \mathbf{V}_i, \mathbf{V}^{[t]}_i\right).$$

Next, fix $(\mathbf{V}_i, \mathbf{V}^{[t]}_i)$. Since $\mathbf{A} \leftrightarrow (\mathbf{Z}'_{i-1}, \mathbf{V}_{i,\mathbf{B}}, \mathbf{V}^{[t]}_{i,\mathbf{B}}, \mathbf{V}_i, \mathbf{V}^{[t]}_i) \leftrightarrow (\mathbf{Q}_{\mathsf{r},i}, \mathbf{Q}^{[t]}_{\mathsf{r},i})$, by Lemma 3.7

$$(\mathbf{W}_{i,\mathbf{A}} \approx_{(13\cdot 2^i-8)\varepsilon} \mathbf{U}_r) \mid \left(\mathbf{Z}'_{i-1}, \mathbf{V}_{i,\mathbf{B}}, \mathbf{V}^{[t]}_{i,\mathbf{B}}, \mathbf{V}_i, \mathbf{V}^{[t]}_i, \mathbf{Q}_{\mathsf{r},i}, \mathbf{Q}^{[t]}_{\mathsf{r},i}\right),$$

which is exactly (4.10). Fix $(\mathbf{Q}_{\mathsf{r},i}, \mathbf{Q}^{[t]}_{\mathsf{r},i})$. Because $(\mathbf{W}_{i,\mathbf{A}}, \mathbf{W}^{[t]}_{i,\mathbf{A}})$ are deterministic functions of $(\mathbf{A}, \mathbf{Q}_{\mathsf{r},i}, \mathbf{Q}^{[t]}_{\mathsf{r},i})$ and $(\mathbf{W}_{i,\mathbf{B}}, \mathbf{W}^{[t]}_{i,\mathbf{B}})$ are deterministic functions of $(\mathbf{B}, \mathbf{Q}_{\mathsf{r},i}, \mathbf{Q}^{[t]}_{\mathsf{r},i})$, we get (4.11). Finally we specify the entropy requirement of $\mathbf{X}, \mathbf{Y}$ for our applications of Lemma 3.7. Observe that every time we apply Lemma 3.7 on $\mathbf{A}$, we condition on some random variables in $\mathbf{Z}_h$, apply an extractor with error ε and output at most r bits. The conditional entropy of $\mathbf{A}$ is at least

$$\widetilde{\mathrm{H}}_\infty(\mathbf{A} \mid \mathbf{Z}_h) \geq \widetilde{\mathrm{H}}_\infty(\mathbf{A} \mid \mathbf{Z}) - (t+1)\cdot O(d_0 + \log(n/\varepsilon) + h(d_y + d_x)),$$

and we need this value to be $O(tr)$. Every time we apply Lemma 3.7 on $\mathbf{Y}$, we condition on some random variables in $\mathbf{Z}_h$, take an extractor from Lemma 2.17 with error ε and output at most d_y bits. The conditional entropy of $\mathbf{Y}$ is at least

$$\widetilde{\mathrm{H}}_\infty(\mathbf{Y} \mid \mathbf{Z}_h) \geq d - (t+1)\cdot O(d_0 + \log(n/\varepsilon) + h(d_y + d_x)),$$

and we need this to be $O(td_y)$.

Since $\mathbf{W}_h = \mathbf{W}_{h,\mathbf{A}} + \mathbf{W}_{h,\mathbf{B}}$, (4.10) and (4.11) together imply

$$(\mathbf{W}_h \approx_{(13t-8)\varepsilon} \mathbf{U}_r) \mid (\mathbf{W}_h^{[t]}, \mathbf{Y}, \mathbf{Y}^{[t]}).$$

Therefore if $m \leq r$, it suffices to output $\mathrm{ACB}(\mathbf{X}, \mathbf{Y}, \alpha) = \mathrm{Prefix}(\mathbf{W}_h, m)$. If $m > r$, we can do one more round of alternating extraction to increase the output length. Let $\mathrm{LExt}_{\mathsf{out}} : \{0,1\}^n \times \{0,1\}^{d_{\mathsf{out}}} \to \{0,1\}^m$ be a linear strong seeded extractor with error ε from Theorem 4.14 (by taking $c = t$) and $\mathrm{Ext}_{\mathsf{out}} : \{0,1\}^d \times \{0,1\}^r \to \{0,1\}^{d_{\mathsf{out}}}$ be a seeded extractor from Lemma 2.17. It suffices to take $d_{\mathsf{out}} = O\left(\frac{m}{t} + \log^2(t+1)\log(\frac{n}{\varepsilon})\right)$. Then

1. Compute $\mathbf{Q}_{\mathsf{out}} := \mathrm{Ext}_{\mathsf{out}}(\mathbf{Y}, \mathbf{W}_h)$.
2. Output $\mathbf{W}_{\mathsf{out}} := \mathrm{LExt}_{\mathsf{out}}(\mathbf{X}, \mathbf{Q}_{\mathsf{out}})$.

Since $(\mathbf{W}_h \approx \mathbf{U}) \mid (\mathbf{Z}_i, \mathbf{W}_{h,\mathbf{B}}, \mathbf{W}_{h,\mathbf{B}}^{[t]})$, $(\mathbf{W}_h, \mathbf{W}_h^{[t]}) \leftrightarrow (\mathbf{Z}_i, \mathbf{W}_{h,\mathbf{B}}, \mathbf{W}_{h,\mathbf{B}}^{[t]}) \leftrightarrow (\mathbf{Y}, \mathbf{Y}^{[t]})$ forms a Markov chain and

$$\widetilde{\mathrm{H}}_\infty(\mathbf{Y} \mid \mathbf{Z}_i, \mathbf{W}_{h,\mathbf{B}}, \mathbf{W}_{h,\mathbf{B}}^{[t]}) \geq d - (t+1) \cdot O(d_0 + \log(n/\varepsilon) + h(d_y + d_x)).$$

If this value is $O(td_{\mathrm{out}})$ then by Lemma 3.7

$$(\mathbf{Q}_{\mathsf{out}} \approx_{(13t-6)\varepsilon} \mathbf{U}_{d_{\mathsf{out}}}) \mid (\mathbf{Q}_{\mathsf{out}}^{[t]}, \mathbf{Z}_i, \mathbf{W}_{h,\mathbf{B}}, \mathbf{W}_{h,\mathbf{B}}^{[t]}, \mathbf{W}_h, \mathbf{W}_h^{[t]}).$$

Now note that $\mathbf{A}$ is independent of $(\mathbf{Q}_{\mathsf{out}}, \mathbf{Q}_{\mathsf{out}}^{[t]})$ conditioned on $(\mathbf{Z}_i, \mathbf{W}_{h,\mathbf{B}}, \mathbf{W}_{h,\mathbf{B}}^{[t]}, \mathbf{W}_h, \mathbf{W}_h^{[t]})$, and

$$\widetilde{\mathrm{H}}_\infty(\mathbf{A} \mid \mathbf{Z}_i, \mathbf{W}_{h,\mathbf{B}}, \mathbf{W}_{h,\mathbf{B}}^{[t]}, \mathbf{W}_h, \mathbf{W}_h^{[t]}) \geq k - (t+1) \cdot O(d_0 + \log(n/\varepsilon) + h(d_y + d_x)).$$

If this value is $O(tm)$, then again by Lemma 3.7 we can conclude that

$$(\mathbf{W}_{\mathsf{out},\mathbf{A}} \approx_{(13t-4)\varepsilon} \mathbf{U}_m) \mid (\mathbf{W}_{\mathsf{out},\mathbf{A}}^{[t]}, \mathbf{Z}_i, \mathbf{W}_{h,\mathbf{B}}, \mathbf{W}_{h,\mathbf{B}}^{[t]}, \mathbf{W}_h, \mathbf{W}_h^{[t]}, \mathbf{Q}_{\mathsf{out}}, \mathbf{Q}_{\mathsf{out}}^{[t]}).$$

Since $\mathbf{W}_{\mathsf{out}} = \mathbf{W}_{\mathsf{out},\mathbf{A}} + \mathbf{W}_{\mathsf{out},\mathbf{B}}$ and $(\mathbf{W}_{\mathsf{out},\mathbf{A}}, \mathbf{W}^{[t]}_{\mathsf{out},\mathbf{A}})$ are independent of $(\mathbf{Y}, \mathbf{Y}^{[t]}, \mathbf{W}_{\mathsf{out},\mathbf{A}}, \mathbf{W}^{[t]}_{\mathsf{out},\mathbf{A}})$ conditioned on $(\mathbf{Z}_i, \mathbf{W}_{h,\mathbf{B}}, \mathbf{W}^{[t]}_{h,\mathbf{B}}, \mathbf{W}_h, \mathbf{W}^{[t]}_h, \mathbf{Q}_{\mathsf{out}}, \mathbf{Q}^{[t]}_{\mathsf{out}})$, we can conclude that

$$(\mathbf{W}_{\mathsf{out}} \approx_{(13t-4)\varepsilon} \mathbf{U}_m) \mid (\mathbf{W}^{[t]}_{\mathsf{out}}, \mathbf{Z}, \mathbf{Y}, \mathbf{Y}^{[t]}),$$

which means $\mathrm{ACB}(\mathbf{X}, \mathbf{Y}, \alpha) = \mathbf{W}_{\mathsf{out}}$ is a strong t-affine correlation breaker with error $O(t\varepsilon)$. Finally we note that it suffices to take

$$d = O(t(d_0 + d_{\mathrm{out}} + h(d_x + d_y))) = O(m + td_0 + t\log^3(t+1)\log(n/\varepsilon))$$

and

$$k = O(tm + d) = O(tm + td_0 + t\log^3(t+1)\log(n/\varepsilon)).$$

□

Chapter 5

Sumset Extractors

In this section, we show how to construct a "sumset extractor" [CL16b], i.e., an extractor for sum of two independent sources, with min-entropy requirement $< \text{polylog}(n)$, and show several applications of this result on randomness extraction. Notably, our sumset extractor implies an explicit extractor for space-s source [KRVZ11] with min-entropy $2s+\text{polylog}(n)$, which is optimal up to an *additive* $\text{polylog}(n)$ term.

5.1 Introduction

In randomness extraction, the ultimate goal is to construct an explicit extractor which works for a class of sources $\mathcal{X}$ that is as general as possible, so that $\mathcal{X}$ presumably contains the sources we are given. In this chapter, we will focus on *sumset*

This chapter is based on the following joint work:

- [CL22] Eshan Chattopadhyay and Jyun-Jie Liao. Extractors for sum of two sources. In *STOC '22: 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1584–1597. ACM, 2022

sources. First we recall the definition of C-sumset sources, with some additional notation for the $C = 2$ case.

Definition 5.1. *We say a source $\mathbf{X} \in \{0,1\}^n$ is a C-*sumset source *with min-entropy k if there exist C independent (n,k)-sources $\{\mathbf{X}_i\}_{i\in[C]}$ such that $\mathbf{X} = \sum_{i=1}^{C} \mathbf{X}_i$. In addition, for the $C = 2$ case, we say $\mathbf{X}$ is a (n,k_1,k_2)-sumset source (or (n,k_1)-sumset source for short if $k_1 = k_2$) if there exist independent (n,k_1)-source $\mathbf{X}_1$ and (n,k_2)-source $\mathbf{X}_2$ such that $\mathbf{X} = \mathbf{X}_1 + \mathbf{X}_2$.*

This class of sources captures and generalizes two central settings in seedless extraction: (i) 2-independent sources setting: given access to independent n-bit sources $\mathbf{X}_1$ and $\mathbf{X}_2$, clearly $(\mathbf{X}_1, \mathbf{X}_2) = (\mathbf{X}_1, 0^n) + (0^n, \mathbf{X}_2)$ is a 2-sumset source (ii) affine source setting: an affine source $\mathbf{X}$ with min-entropy k can be written as the sum of two independent affine sources $\mathbf{X}_1, \mathbf{X}_2$, each with min-entropy k.[8] Thus, an extractor for the sum of two sources directly gives a *two-source extractor* as well as an *affine extractor*. In addition, a sumset extractor can also extract from other classes of sources interleaved sources [RY11] and small-space sources [KRVZ11] (see Section 5.1.1 for the definitions.)

However, it has been challenging to construct extractors for 2-sumset source with low min-entropy. The only known extractors for the sum of two independent sources before our work is the Paley graph extractor [CG88], which is a $(n, 0.51n, O(\log(n)))$-sumset extractor based on character sum estimates obtained by Karatsuba [Kar71, Kar91]. In [CL16b], Chattopadhyay and Li constructed an explicit extractor for C-sumset sources with min-entropy $k = \mathrm{polylog}(n)$, but their analysis only works for a constant C that is much larger than 2.

[8] For example, we can pick any $b \in \mathrm{Supp}(\mathbf{X})$ and take the distributions of $\mathbf{X}_1, \mathbf{X}_2$ to be the same as $\mathbf{X}$ and $b + \mathbf{X}$ respectively.

In this chapter, we give the first explicit construction of extractors for 2-sumset sources with min-entropy $\mathrm{polylog}(n)$. Formally, we prove the following theorem.

Theorem 5.2. *There exists a universal constant C such that for every $k \geq \log^C(n)$, there exists an explicit extractor $\mathrm{Ext} : \{0,1\}^n \to \{0,1\}^m$ for (n,k)-sumset source with error $n^{-\Omega(1)}$ and output length $m = k^{\Omega(1)}$.*

We can further lower the entropy requirement to almost logarithmic at the expense of worse error parameter of the extractor.

Theorem 5.3. *For every constant $\varepsilon > 0$, there exists a constant C_ε such that there exists an explicit extractor $\mathrm{Ext} : \{0,1\}^n \to \{0,1\}$ with error ε for (n,k)-sumset source where*

$$k = C_\varepsilon \log(n) \log\log(n) \log\log\log^3(n).$$

5.1.1 Applications of Sumset Extractors

In this section we show two applications of our sumset extractors. The first one is an improved entropy bound for *interleaved-source extractors*. Interleaved sources are a natural generalization of independent sources, introduced by Raz and Yehudayoff [RY11] where they called it as "mixed sources". An interleaved source consists of bits from two independent sources which are interleaved in an unknown (but fixed) order. Such sources naturally arise in a scenario that the bits of the input source are communicated remotely to the extractor from two independent sources. In addition, Raz and Yehudayoff [RY11] also showed that an explicit extractor for such sources yields a lower bound in best-partition communication complexity model. The formal definition of interleaved sources is as follows. (For a n-bit string w and a permutation $\sigma : [n] \to [n]$, we use w_σ to denote the string such that the $\sigma(i)$-th bit of w_σ is exactly the i-th bit of w.)

Definition 5.4. *Let $\mathbf{X}_1$ be an (n, k_1)-source, $\mathbf{X}_2$ be an (n, k_2)-source independent of $\mathbf{X}_1$ and $\sigma : [2n] \to [2n]$ be a permutation. Then we say $(\mathbf{X}_1 \circ \mathbf{X}_2)_\sigma$ is an* (n, k_1, k_2)-interleaved sources *(or (n, k_1)-*interleaved sources *for short if $k_1 = k_2$.)*

Raz and Yehudayoff [RY11] constructed an extractor for $(n, (1-\beta)n)$-interleaved sources with $2^{-\Omega(n)}$ error for a small constant $\beta > 0$. Subsequently, Chattopadhyay and Zuckerman [CZ16] constructed an extractor for $(n, (1-\gamma)n, O(\log(n)))$-interleaved sources with error $n^{-\Omega(1)}$ for a small constant $\gamma > 0$. A recent work by Chattopadhay and Li [CL20] gave an extractor for $(n, (2/3+\delta)n)$-interleaved sources with error $2^{-n^{\Omega(1)}}$, where δ is an arbitrarily small constant. In summary, all prior work required at least one of the sources to have min-entropy at least $0.66n$.

It was observed in [CL16b] that an (n, k)-interleaved source is also a $(n, k, 2)$-sumset sources because $(\mathbf{X}_1 \circ \mathbf{X}_2)_\sigma = (\mathbf{X}_1 \circ 0^n)_\sigma + (0^n \circ \mathbf{X}_2)_\sigma$. Therefore, with our extractors for sum of two sources, we obtain the first extractors for interleaved two sources with polylogarithmic entropy.

Corollary 5.5. *There exists a universal constant C such that for every $k \geq \log^C(n)$, there exists an explicit extractor $\mathrm{Ext} : \{0,1\}^n \to \{0,1\}^m$ for (n, k)-interleaved sources with error $n^{-\Omega(1)}$.*

Corollary 5.6. *For every constant $\varepsilon > 0$, there exists a constant C_ε and an explicit extractor $\mathrm{Ext} : \{0,1\}^n \to \{0,1\}$ with error ε for (n, k)-interleaved sources where*

$$k = C_\varepsilon \log(n) \log\log(n) \log\log\log^3(n).$$

The second application is an extractor for small-space sources with almost optimal entropy requirement. A small-space source is a distribution that is sampled by algorithms with limited memory, which was formally defined by Kamp, Rao, Vadhan and Zuckerman [KRVZ11] as follows.

Definition 5.7. *A space-s sampling procedure $\mathcal{A}$ with n-bit output is defined as follows. For every (i, j) s.t. $i \in \mathbb{Z}, 0 \leq i < n$ and $j \in \{0,1\}^s$, let $\mathcal{D}_{i,j}$ be a distribution over $\{0,1\} \times \{0,1\}^s$. Then $\mathcal{A}$ maintains an internal state $\mathsf{st} \in \{0,1\}^s$, which is initially 0^s, and runs the following steps for time step i from 0 to $n-1$:*

1. *Sample $(x_{i+1}, \mathsf{nextSt}) \in \{0,1\} \times \{0,1\}^s$ from $\mathcal{D}_{i,\mathsf{st}}$.*
2. *Output x_{i+1}, and assign $\mathsf{st} := \mathsf{nextSt}$.*

Furthermore, the distribution $\mathbf{X}$ of the output $(x_1, \ldots, x_n)$ is called a space-s source.

In [KRVZ11], Kamp, Rao, Vadhan and Zuckerman constructed an extractor for space-s source with entropy $k \geq Cn^{1-\gamma}s^{\gamma}$ with error $2^{-n^{\Omega(1)}}$, for a large enough constant C and a small constant $\gamma > 0$. Chattopadhyay and Li [CL16b] then constructed an extractor with error $n^{-\Omega(1)}$ for space-s source with entropy $k \geq s^{1.1}2^{\log^{0.51}(n)}$ based on their sumset extractors. Recently, based on a new reduction to affine extractors, Chattopadhyay and Goodman [CG21] improved the entropy requirement to $k \geq s \cdot \mathrm{polylog}(n)$ (or $k \geq s\log^{2+o(1)}(n)$ in the constant error setting).[9]

With our new extractors for the sum of two sources, we can use the reduction in [CL16b] to get extractors for space-s source with entropy $s\log(n) + \mathrm{polylog}(n)$, which is already an improvement over the result in [CG21]. In this chapter, we further improve the reduction and obtain the following results.

Theorem 5.8. *There exists a universal constant C such that for every s and every $k \geq 2s + \log^C(n)$, there exists an explicit extractor $\mathrm{Ext} : \{0,1\}^n \to \{0,1\}^m$ with error $n^{-\Omega(1)}$ and output length $m = (k-2s)^{\Omega(1)}$ for space-s sources with min-entropy k.*

[9] Here we focus on the small-space extractors which minimize the entropy requirement. For small-space extractors with negligible error, the best known extractor requires min-entropy $\approx n^{0.51}s^{0.49}$ [CG21].

Theorem 5.9. *For every constant $\varepsilon > 0$, there exists a constant C_ε such that there exists an explicit extractor* $\mathrm{Ext} : \{0,1\}^n \to \{0,1\}$ *with error ε for space-s sources with min-entropy*

$$2s + C_\varepsilon \log(n) \log\log(n) \log\log\log^3(n).$$

Interestingly, the entropy requirement of our extractors has optimal dependence on the space parameter s, as Kamp, Rao, Vadhan and Zuckerman [KRVZ11] showed that it is impossible to construct an extractor for space-s source with min-entropy $\leq 2s$. Moreover, the min-entropy requirement in Theorem 5.9 almost matches the non-constructive extractor in [KRVZ11] that requires min-entropy at least $2s + O(\log(n))$.

5.1.2 Other Results

In this chapter we also show two other results related to sumset extractors, which might be of independent interest. The first one is the fact that the inner product function (i.e., the Hadamard extractor [Vaz87, CG88]) is also an extractor for sum of two n-bit sources with min-entropy k_1 and k_2 as long as $k_1 + k_2 > n$. The proof is along the lines of a standard proof which shows that the inner product function is a two-source extractor. However, to the best of our knowledge this result was not known before.

Theorem 5.10. *Let* $\mathsf{IP} : \{0,1\}^n \to \{0,1\}$ *be the inner production function* $\mathsf{IP}(x) := \sum_{i \in [n/2]} x_i x_{i+n/2}$, *where x_i denotes the i-th bit of x. Then for any $k_1 + k_2 > n$,* IP *is an extractor for (n, k_1, k_2)-sumset source with error $\varepsilon = 2^{\frac{n-(k_1+k_2)}{2}-1}$.*

The second result is that a sumset source with small "doubling constant" is close to a convex combination of affine sources. The formal definition of doubling constant is as follows. First we recall the definition of sumsets and related notation.

Definition 5.11 (sumsets). *For any sets $A, B \subseteq \mathbb{F}_2^n$, we use $A + B$ to denote $\{a + b : a \in A, b \in B\}$, $2A$ to denote $A + A$, and ℓA to denote $(\ell - 1)A + A$ for any integer $\ell > 2$.*

Definition 5.12. *For $A, B \subseteq \mathbb{F}_2^n$ s.t. $|A| = |B|$ we define the doubling constant of (A, B) to be $|A + B| / |B|$, and the doubling constant of A to be $|2A| / |A|$.*

The study of doubling constant is motivated by the fact that (A, B) has doubling constant $r = 1$ if and only if A and B are cosets of the same subspace. A central research topic in additive combinatorics (see [TV06]) is to study how well A and B are approximated by an affine subspace when the doubling constant is some small value $r > 1$. The best-known result in this field is perhaps the "polynomial Freiman-Ruzsa conjecture" by Marton (see [Ruz99]), which states that A and B can be covered by $\mathrm{poly}(r)$ translates of the same subspace. This result was recently proved by Gowers, Green, Manners and Tao [GGMT23]. Another famous result in this area is the Bogolyubov-Ruzsa lemma, which states that if (A, A) has small doubling constant r then $4A$ fully contains a subspace of dimension $\log(|A|) - C_r$ for some constant C_r that depends on r. The strongest bound for this lemma is $C_r = O(\log^4(r))$ by Sanders [San12].

Toward proving the $C_r = O(\log^4(r))$ bound of Bogolyubov-Ruzsa lemma, Sanders also proved an "approximate Bogolyubov-Ruzsa lemma" which states that $A + B$ contains $1 - \gamma$ fraction of an affine subspace with dimension $\log(|A|) - O_\gamma(\log^4(r))$ for any constant $\gamma > 0$. We prove a statistical version of this result.

Theorem 5.13. *Let $\mathbf{A}, \mathbf{B}$ be uniform distribution over $A, B \subseteq \mathbb{F}_2^n$ s.t. $|A| = |B| = 2^k$ and $|A + B| \leq r\,|A|$. Then $\mathbf{A} + \mathbf{B}$ is ε-close to a convex combination of affine sources with entropy $k - O(\varepsilon^{-2} \log(r) \log^3(r/\varepsilon))$.*

This implies the following corollary.

Corollary 5.14. *Let* $\mathbf{A}, \mathbf{B}$ *be uniform distribution over* $A, B \subseteq \mathbb{F}_2^n$ *s.t.* $|A| = |B| = 2^k$ *and* $|A + B| \leq r\,|A|$. *If* $\mathrm{AffExt} : \{0,1\}^n \to \{0,1\}^m$ *is an extractor for affine sources with entropy* $k - O(\log^4(r))$, *then* $\mathrm{AffExt}(\mathbf{A} + \mathbf{B})$ *is* $O(1)$*-close to* $\mathbf{U}_m$.

Subsequent works. As we mentioned in Chapter 4, Li [Li23] recently constructed a correlation breaker with optimal parameter for $t = O(1)$. This also implies a sumset extractor with constant error for $O(\log(n))$ min-entropy, and the entropy requirement of constant-error extractors for interleaved sources and small-space sources can also be improved to $O(\log(n))$ and $2s + O(\log(n))$ respectively.

Organization. In Section 5.2 we give an overview of our sumset extractor construction, and the improved reduction from small-space extractor to sumset extractors. In Section 5.3 we formally prove our results on sumset extractors, Theorem 5.2 and Theorem 5.3. In Section 5.4 we prove our small-space extractor results, Theorem 5.9 and Theorem 5.8. In Section 5.5 we prove Theorem 5.10. In Section 5.6 we prove Theorem 5.13.

5.2 Proof Overview

In this section we give a proof overview of our results on sumset extractors (Theorem 5.2 and Theorem 5.3) and small-space extractors (Theorem 5.8 and Theorem 5.9).

5.2.1 Decoupling Errors with a Sampler

Our construction of sumset extractors also follows the Chattopadhyay-Zuckerman framework, but here we need to deal with an issue that we ignored in Chapter 3: the output of a strong seeded extractor is not exactly uniform. First we give an overview of how this issue is resolved in the setting of two-source extractors. Recall that in Section 3.2, our first step to construct an NOBF source with two independent sources $\mathbf{X}, \mathbf{Y}$ is to take a strong seeded extractor Ext with error ε, enumerate all $D = 2^d$ seeds $i \in \{0,1\}^d$ and compute $\mathbf{Y}_i := \mathrm{Ext}(\mathbf{Y}, i)$. In the second step we use $\mathbf{X}$ to break correlation and get $\mathbf{Z}_i = \mathrm{CB}(\mathbf{X}, \mathbf{Y}_i, i)$ for each i, and we claimed that we get a (q,t)-NOBF source $\mathbf{Z} := (\mathbf{Z}_i)_{i\in[t]}$ of D bits for some $q = D^{1-\Omega(1)}$. However, because each "good" $\mathbf{Y}_i$ is only ε-close to uniform, after applying the correlation breaker we also only have the guarantee that each t good bits $\mathbf{Z}_i$ is $O(\varepsilon)$-close to uniform. When we apply Lemma 2.4 to show that $\mathbf{Z}$ is close to a (q,t)-NOBF source, the error bound we get is $O(D^t\varepsilon)$. However, because $D > (1/\varepsilon)^2$ ([RT00]), the $O(D^t\varepsilon)$ bound is actually a trivial bound that is greater than 1.

Based on the discussion above, we can see that we have different tolerance for two different types of error: for the sequence $(\mathbf{Y}_1, \ldots, \mathbf{Y}_D)$, suppose that for all except for ε_1 fraction of $i \in [D]$, $\mathbf{Y}_i$ is ε_2-close to uniform. If we want the later steps to work, for the first type of errors ε_1 we can tolerate $\varepsilon_1 = D^{-\gamma}$ for some small constant $\gamma < 1$, but for the second type of errors ε_2 we need $\varepsilon_2 < D^{-t}$ in order to get a non-trivial result in the end, and this is not possible with our original approach. In addition, we note that the BDT error reduction [BDT19] we saw in Section 3.5 only gives a way to reduce ε_1 but does not change ε_2.

To decouple ε_2 from D, Li [Li15] developed the idea of using a sampler. For parameters $\varepsilon_2 \ll \varepsilon_1$, first take a strong seeded extractor Ext with error ε_2 and enu-

merate all seeds to generate a sequence of random variables that are mostly ε_2-close to uniform. Then take another independent source as the randomness to apply a sampler for weak sources (Lemma 2.19) and sample D random variables from the sequence, so that all except for ε_1 fraction of the samples are "good" (ε_2-close to uniform). Now D only depends on ε_1 and is decoupled from ε_2, so it becomes possible to take $\varepsilon_2 < D^{-t}$. This already consumes two independent sources, but Li showed that one can recycle one of the sources to break correlation with a more careful analysis.

In [CZ19], Chattopadhyay and Zuckerman showed a more modular way to apply this trick, and this is also the approach that we adopt in this chapter. Roughly speaking, this approach utilizes the strongness of a correlation breaker to show that for any parameters D, t, if one pick CB to be a (t, k, ε_2)-strong correlation breaker for small enough error ε_2, then most of the seed y for CB is "good", where "good" means the collection of any set $\{y_1, \ldots, y_D\}$ with $(D-q)$ good y satisfies that $(\mathrm{CB}(\mathbf{X}, y_1, 1), \ldots, \mathrm{CB}(\mathbf{X}, y_D, D))$ is close to a (q, t)-NOBF source. Then one can use the second source $\mathbf{Y}$ to pick $D = \mathrm{poly}(1/\varepsilon_1)$ samples of y with all but ε_1 fraction being good, and get a $(\varepsilon_1 D, t)$-NOBF source.[10] Note that by the fact that an extractor is also a sampler for weak sources (Lemma 2.19), the final NOBF source is of the form $(\mathrm{CB}(\mathbf{X}, \mathrm{Ext}(\mathbf{Y}, i), i))_{i \in [D]}$, which is exactly the same as what we described in Section 3.2. However, the analysis does not follow our intuitive description in Section 3.2.

[10] The original approach in [CZ19] considers non-malleable extractors [DW09] instead of correlation breakers, but the idea is very similar.

5.2.2 Sampling with Sumset Sources

Now we move to the setting of 2-sumset sources. Suppose we are given a 2-sumset source $\mathbf{X} = \mathbf{X}_1 + \mathbf{X}_2$ and want to construct an NOBF source from $\mathbf{X}$. Our construction will be similar to what we did in Chapter 4: take a strong linear seeded extractor LExt and an affine correlation breaker ACB, then enumerate all the seeds and compute $(\mathrm{ACB}(\mathbf{X}, \mathrm{LExt}(\mathbf{X}, i), i))_{i\in[D]}$.[11] However, the analysis is very different.

If we again ignore the fact that each good $\mathrm{LExt}(\mathbf{X}, i)$ is not exactly uniform, then we can get a "proof" that is essentially what we saw in Chapter 4. That is, suppose that $\mathrm{LExt}(\mathbf{X}_2, i_1), \mathrm{LExt}(\mathbf{X}_2, i_2), \ldots, \mathrm{LExt}(\mathbf{X}_2, i_t)$ are exactly uniform. If we condition on $\mathrm{LExt}(\mathbf{X}_1, i_1), \ldots, \mathrm{LExt}(\mathbf{X}_1, i_t)$, then $\mathbf{X}_1$ still has some entropy left. In addition, each $\mathrm{LExt}(\mathbf{X}, i_j) = \mathrm{LExt}(\mathbf{X}_1, i_j) + \mathrm{LExt}(\mathbf{X}_2, i_j)$ is still uniform but becomes independent of $\mathbf{X}_1$. Therefore we can follow the same analysis as in Chapter 4. However, if we take the error of each $\mathrm{LExt}(\mathbf{X}_2, i_j)$ into consideration, then we face a similar issue as what we saw in the last section. In fact, the C-sumset extractor result by Chattopadhyay and Li [CL16b] is also based on this proof outline. To reduce the error of $\mathrm{LExt}(\mathbf{X}_2, i_j)$, they further assumed that $\mathbf{X}_2$ is the sum of $C-1$ independent sources for some large C.[12]

If we want to get an extractor for 2-sumset sources, then we need to apply the sampling trick as for the case of two independent sources. However, things become tricky in the setting of sumset sources: whether a seed y is good depends on $\mathbf{X}$, so this event is now correlated with the randomness $\mathbf{X}$ we use for sampling. In this case there is no guarantee that the sampler works properly. It was conjectured in [CL16b] that the linearity of the sampler LExt should help, but they didn't find a

[11] To extract from sources with almost logarithmic entropy (Theorem 5.3), we need to apply the BDT error reduction (Section 3.5), and the construction changes correspondingly.

[12] Recall that in the extreme $C = \infty$ case, where $\mathbf{X}_2$ is affine, $\mathrm{LExt}(\mathbf{X}_2, i_j)$ has 0 error.

way to analyze it. In this chapter we will show that this intuition is correct.

As a warm up, suppose we have a strong seeded extractor Ext with error ε_1, and we want to use a source $\mathbf{X} = \mathbf{X}_1 + \mathbf{X}_2$ to sample some good seeds y such that $\mathrm{Ext}(\mathbf{X}, y)$ is uniform. Our goal is to show that we can still view this reduction as if we were sampling good seeds with $\mathbf{X}_2$ and using these seeds to extract from $\mathbf{X}_1$. Let Samp denote a sampler for weak sources, and consider the i-th sampled output, $\mathrm{Ext}(\mathbf{X}, \mathrm{Samp}(\mathbf{X}, i))$. Our main observation is, if $\mathrm{Samp}(\cdot, i)$ is a linear function, then we can assume that we compute $\mathrm{Ext}(\mathbf{X}, \mathrm{Samp}(\mathbf{X}, i))$ in the following steps:

1. First sample $x_2 \sim \mathbf{X}_2$.
2. Use x_2 as the randomness of Samp to sample a "seed" $s := \mathrm{Samp}(\mathbf{X}_2, i)$.
3. Output $\mathrm{Ext}'_{x_2,i}(\mathbf{X}_1, s) := \mathrm{Ext}(\mathbf{X}_1 + x_2, s + \mathrm{Samp}(\mathbf{X}_1, i))$.

First we claim that $\mathrm{Ext}'_{x_2,i}$ is also a strong seeded extractor. To see why this is true, observe that if we fix $\mathrm{Samp}(\mathbf{X}_1, i) = \Delta$, then $\mathrm{Ext}'_{x_2,i}(\mathbf{X}_1, \mathbf{U}) = \mathrm{Ext}(\mathbf{X}_1 + x_2, \mathbf{U} + \Delta)$. As long as $\mathbf{X}_1$ still has enough entropy after fixing $\mathrm{Samp}(\mathbf{X}_1, i)$, Ext works properly since $\mathbf{X}_1 + x_2$ is independent of $\mathbf{U} + \Delta$, $\mathbf{X}_1 + x_2$ still has enough entropy and $\mathbf{U} + \Delta$ is also uniform. Therefore, we can use $\mathrm{Ext}'_{x_2,i}$ and $\mathbf{X}_1$ to define a set of good seeds s which make $\mathrm{Ext}'_{x_2,i}(\mathbf{X}_1, s)$ close to uniform, and most of the seeds should be good. Then we can equivalently view the sampling step as if we were sampling good seeds for $\mathrm{Ext}'_{x_2,i}$ using $\mathbf{X}_2$ as the randomness.

There are still two problems left. First, the definition of $\mathrm{Ext}'_{x_2,i}$ depends on x_2, which is the randomness we use for sampling. To solve this problem, we can take Ext to be linear, and prove that $(1 - \sqrt{\varepsilon_1})$ fraction of the seeds s are good in the sense that $\mathrm{Ext}'_{x_2,i}(\mathbf{X}_1, s)$ is close to uniform for *every* x_2. Second, $\mathrm{Ext}'_{x_2,i}$ depends on i, which is the index of our samples. Similarly we change the definition of good

seeds so that a seed s is good if $\mathrm{Ext}'_{x_2,i}(\mathbf{X}_1, s)$ is good for every x_2 and i, and by union bound we can show that $(1 - 2^{d_2}\sqrt{\varepsilon_1})$ fraction of the seeds are good. As long as $\varepsilon_1 \ll 2^{-2d_2}$, most of the seeds should be good. Now the definition of good seeds is decoupled from the sampling step, and thus we can show that most of the sampled seeds are good.

Next we turn to the case of t-correlation extractors. First we define a good seed with respect to every possible "$(t+1)$-local view". We say a seed s is good with respect to x_2 and a set of indices $T = \{i_0, i_1, \ldots, i_t\}$ if for every $s^1, \ldots, s^t \in \{0,1\}^{d_1}$,

$$(\mathrm{CB}(\mathbf{X}_1 + x_2, s + \mathrm{Samp}(\mathbf{X}_1, i_0), i_0) \approx_{\sqrt{\varepsilon_1}} \mathbf{U}_1$$

conditioned on

$$\{\mathrm{CB}(\mathbf{X}_1 + x_2, s^j + \mathrm{Samp}(\mathbf{X}_1, i_j), i_j)\}_{j \in [t]}.$$

We claim that if $\mathbf{X}_1$ has enough entropy when conditioned on $\{\mathrm{Samp}(\mathbf{X}_1, i)\}_{i \in T}$, then $1 - \sqrt{\varepsilon_1}$ of the seeds are good with respect to x_2 and T. If we can prove that most of the seeds s we sample using $\mathbf{X}_2$ are good with respect to x_2 and every set of indices T, then the we can conclude that the output $\mathbf{R}_2 = (\mathrm{CB}(\mathbf{X}, \mathrm{Samp}(\mathbf{X}, i), i))_{i \in \{0,1\}^{d_2}}$ is $D_2^{t+1}\sqrt{\varepsilon_1}$-close to an NOBF source.

Next we need to show that most of the seeds are good with respect to *every* x_2 and T, so that the sampling step is decoupled from the definition of good seeds. To deal with the dependency on T, we simply take the union bound over T and show that $1 - D_2^{t+1}\sqrt{\varepsilon_1}$ of the seeds are good. To deal with the dependency on x_2, we claim that it suffices to replace CB with ACB.

5.2.3 Improved Reduction for Small-space Extractors

In this section we give an overview of our new reduction from small-space sources to sumset sources, and compare it with previous reductions. As in all the previous works on small-space source extractors, our reduction is based on a simple fact: conditioned on the event that the sampling procedure is in state j at time i, the small-space source $\mathbf{X}$ can be divided into two independent sources $\mathbf{X}_1 \in \{0,1\}^i, \mathbf{X}_2 \in \{0,1\}^{n-i}$, such that $\mathbf{X}_1$ contains the bits generated before time i, and $\mathbf{X}_2$ contains the bits generated after time i. Kamp, Rao, Vadhan and Zuckerman [KRVZ11] proved that if we pick some equally distant time steps $i_1, \ldots, i_{\ell-1}$ and condition on the states visited at these time steps, we can divide the small-space source into ℓ independent blocks such that some of them have enough entropy. However, such a reduction does not work for entropy smaller than $\sqrt{n}$ (cf. [CG21]).

To break this $\sqrt{n}$ barrier, Chattopadhyay and Li [CL16b] observed that with a sumset source extractor we can extract from the concatenation of independent sources with *unknown and uneven length*. They then showed that with a sumset source extractor, we can *adaptively* pick which time steps to condition on depending on the given source. Chattopadhyay and Goodman [CG21] further refined this reduction and showed how to improve the entropy requirement by reducing to a convex combination of affine sources. The reductions in [CL16b] and [CG21] can be viewed as "binary searching" the correct time steps to condition on, so that the given source $\mathbf{X}$ becomes the concatenation of independent blocks $(\mathbf{X}_1, \ldots, \mathbf{X}_{O(\log(n))})$ such that some of them have enough entropy. However, even though with our extractors for sum of two sources we only need two of the blocks to have enough entropy, the "binary-search-based" reduction would condition on at least $\log(n)$ time steps and waste $s \log(n)$ entropy.

A possible way to improve this reduction is by directly choosing the "correct" time step to condition on so that we only get two blocks $\mathbf{X}_1 \circ \mathbf{X}_2$ both of which have enough entropy. However this is not always possible. For example, consider a distribution which is a convex combination of $\mathbf{U}_{n/2} \circ 0^{n/2}$ and $0^{n/2} \circ \mathbf{U}_{n/2}$. This distribution is a space-1 source and has entropy $n/2$, but no matter which time step we choose to condition on, one of the two blocks would have zero entropy.

To solve these problems, we will not condition on a fixed time step, but carefully define some events to condition on instead. For a space-s source and its sampling process, we define (i, j) to be a "stopping state" such that if our sampling process has internal state j at time i, then the distribution of the remaining $n - i$ bits has min-entropy *at most* $k + s + \log(n/\varepsilon)$. Then we condition on the *first stopping state* passed by the sampling process. We claim that if such a state is (i, j), then if we take $\mathbf{X}_1$ to be the first i output bits and $\mathbf{X}_2$ to be the remaining $n - i$ bits, then with high probability both $\mathbf{X}_1$ and $\mathbf{X}_2$ have min-entropy at least k as long as the space-s source has min-entropy $2(k + s + \log(n/\varepsilon))$.

To see why this is the case, for every time-state pair (i, j) let $\mathbf{X}_{(i,j)}$ denote the min-entropy. We claim that whenever we move from $(i-1, j')$ to $(i+1, j)$ in the sampling process, the min-entropy of $\mathbf{X}_{(i,j)}$ can only be lower than $\mathbf{X}_{(i-1,j')}$ by at most $s + \log(n/\varepsilon)$, except with probability ε. Therefore $\mathbf{X}_{(i,j)}$ should have min-entropy at least k if (i, j) is the *first* stopping state. In addition, because $\mathbf{X}_2 = \mathbf{X}_{(i,j)}$ has min-entropy at least $k + s + \log(n/\varepsilon)$, and with probability at least $1 - \varepsilon$ the total min-entropy of $\mathbf{X}_1, \mathbf{X}_2$ is at least $2k + s + \log(n/\varepsilon)$ by chain-rule, so the min-entropy of $\mathbf{X}_1$ should also be at least k.

5.3 Sumset Extractors

In this section we formally prove our main sumset extractor results (Theorem 5.2 and Theorem 5.3). The construction of our extractors relies on the following lemma:

Lemma 5.15 (main lemma). *For every constant $\gamma < 1$ and every $t \in \mathbb{N}$, there exists $N = n^{O(1)}$ and an explicit function $\mathrm{Reduce} : \{0,1\}^n \to \{0,1\}^N$ s.t. for every (n,k)-sumset source $\mathbf{X}$, where*

$$k = O\left(t^3 \log(n) \cdot \left(\frac{\log\log(n)}{\log\log\log(n)} + \log^3(t)\right) \cdot \left(\log\log\log^4(n) + \log^4(t)\right)\right),$$

$\mathrm{Reduce}(\mathbf{X})$ is $N^{-\gamma}$-close to a $(N^{1-\gamma}, t)$-NOBF source.

Before we prove Lemma 5.15, first we show how to prove Theorem 5.2 and Theorem 5.3 based on Lemma 5.15.

Proof of Theorem 5.2. Let $\mathrm{Reduce} : \{0,1\}^n \to \{0,1\}^N$ be the function from Lemma 5.15 by taking $\gamma = 0.1$. Note that $N = \mathrm{poly}(n)$. Let $\mathrm{BFExt} : \{0,1\}^N \to \{0,1\}^m$ be the NOBF-source extractor from Lemma 3.3. Let $\mathbf{X}$ be a (n,k)-sumset source, where k is defined later. If $\mathrm{Reduce}(\mathbf{X})$ is $N^{-\Omega(1)}$-close to a $(N^{0.9}, t)$-NOBF source where $t = (m\log(N))^{C_{3.3}}$, then

$$\mathrm{Ext}(\mathbf{X}) := \mathrm{BFExt}(\mathrm{Reduce}(\mathbf{X}))$$

is $n^{-\Omega(1)}$-close to uniform. By Lemma 5.15 it suffices to take

$$k = O(t^3 \log^7(t)\log(n)) \le (m\log(n))^{1+3C_{3.3}}.$$

□

Proof of Theorem 5.3. Let $\mathrm{Reduce} : \{0,1\}^n \to \{0,1\}^N$ be the function from Lemma 5.15 by taking $\gamma = 0.6$. Note that $N = \mathrm{poly}(n)$. Let $\mathrm{Maj} : \{0,1\}^N \to \{0,1\}$ be the

NOBF-source extractor from Lemma 3.2, i.e., the majority function. Let $\mathbf{X}$ be a (n,k)-sumset source, where k is defined later. If $\mathrm{Reduce}(\mathbf{X})$ is $(\varepsilon/2)$-close to a $(N^{0.4},t)$-NOBF source where $t = O(\varepsilon^{-2}\log^2(1/\varepsilon)) = O(1)$, then

$$\mathrm{Ext}(\mathbf{X}) := \mathrm{Maj}(\mathrm{Reduce}(\mathbf{X}))$$

is ε-close to uniform. By Lemma 5.15 it suffices to take

$$k = O(\log(n)\log\log(n)\log\log\log^3(n)).$$

□

Next we prove Lemma 5.15. First we prove the following lemma, which is an analog of [CZ19, Lemma 2.17]. Roughly speaking, we show that even if the seeds of the correlation breaker are added by some leakage from the source, most of the seeds are still good.

Lemma 5.16. *For every error parameter $\gamma > 0$ the following holds. Let*

- $\mathrm{ACB} : \{0,1\}^n \times \{0,1\}^d \times \{0,1\}^a \to \{0,1\}^m$ *be a (t,k,ε)-strong affine correlation breaker*
- $L : \{0,1\}^n \times \{0,1\}^a \to \{0,1\}^d$ *be any deterministic function, which we call the* leakage function
- $\alpha, \alpha^{[t]}$ *be any a-bit* advice *s.t. $\alpha \neq \alpha^i$ for every $i \in [t]$*
- $\mathbf{A}$ *be an $(n, k+(t+1)\ell+\log(1/\varepsilon))$-source*

For every $b \in \{0,1\}^n$, $y \in \{0,1\}^d$, define

$$\mathbf{R}_{b,y} := \mathrm{ACB}(\mathbf{A}+b, y+L(\mathbf{A},\alpha), \alpha)$$

and for every $i \in [t]$ define

$$\mathbf{R}^i_{b,y} := \mathrm{ACB}(\mathbf{A} + b, y + L(\mathbf{A}, \alpha^i), \alpha^i).$$

Let $\mathsf{BAD}_{\alpha,\alpha^{[t]}}$ *be the set of "bad seeds", which is defined as*

$$\left\{y \in \{0,1\}^d : \exists b, y^{[t]} \text{ s.t. } (\mathbf{R}_{b,y} \not\approx_\gamma \mathbf{U}_m) \mid \{\mathbf{R}^i_{b,y^i}\}_{i\in[t]}\right\}.$$

Then

$$\Pr_{y\sim\mathbf{U}_d}\left[y \in \mathsf{BAD}_{\alpha,\alpha^{[t]}}\right] \leq 2\varepsilon/\gamma.$$

Proof. Define deterministic functions $f^1, \ldots, f^t : \{0,1\}^d \to \{0,1\}^d$ and $g : \{0,1\}^d \to \{0,1\}^n$ s.t. for every $y \in \mathsf{BAD}_{\alpha,\alpha^{[t]}}$,

$$\left(\mathbf{R}_{g(y),y} \not\approx_\gamma \mathbf{U}_m\right) \mid \left(\{\mathbf{R}^i_{g(y),f^i(y)}\}_{i\in[t]}\right).$$

For $y \notin \mathsf{BAD}_{\alpha,\alpha^{[t]}}$ the values of $f^1(y), f^2(y), \ldots, f^t(y), g(y)$ are defined arbitrarily. Note that the existence of $f^1, \ldots, f^t, g$ is guaranteed by the definition of $\mathsf{BAD}_{\alpha,\alpha^{[t]}}$. Let $\mathbf{W} := \mathbf{U}_d$ and $\delta := \Pr\left[\mathbf{W} \in \mathsf{BAD}_{\alpha,\alpha^{[t]}}\right]$. Observe that

$$(\mathbf{R}_{g(\mathbf{W}),\mathbf{W}} \not\approx_{\gamma\delta} \mathbf{U}_m) \mid (\{\mathbf{R}^i_{g(\mathbf{W}),f^i(\mathbf{W})}\}_{i\in[t]}, \mathbf{W}).$$

Now define $\mathbf{Y} := \mathbf{W} + L(\mathbf{A}, \alpha)$, $\mathbf{Y}^i := \mathbf{W} + L(\mathbf{A}, \alpha^i)$ for every $i \in [t]$ and $\mathbf{B} := g(\mathbf{W})$. Let $\mathbf{Z} := (L(\mathbf{A}, \alpha), L(\mathbf{A}, \alpha^1), \ldots, L(\mathbf{A}, \alpha^t))$. Note that $\mathbf{Z} \in \{0,1\}^{(t+1)\ell}$ is a deterministic function of $\mathbf{A}$. With these new definitions the above equation can be rewritten as

$$(\mathrm{ACB}(\mathbf{A} + \mathbf{B}, \mathbf{Y}, \alpha) \not\approx_{\gamma\delta} \mathbf{U}_m) \mid (\{\mathrm{ACB}(\mathbf{A} + \mathbf{B}, \mathbf{Y}^i, \alpha^i\}_{i\in[t]}, \mathbf{W}). \tag{5.1}$$

Next, observe that the following conditions hold:

- $\widetilde{\mathrm{H}}_\infty(\mathbf{A} \mid \mathbf{Z}) \geq k + \log(1/\varepsilon)$ (by Lemma 2.10)

- $(\mathbf{Y}, \mathbf{Z}) = (\mathbf{U}_d, \mathbf{Z})$.
- $\mathbf{A} \leftrightarrow \mathbf{Z} \leftrightarrow (\mathbf{B}, \mathbf{Y}, \mathbf{Y}^{[t]})$ is a Markov chain.

Note that the last condition holds because $\mathbf{Z}$ is a deterministic function of $\mathbf{A}$, which implies $\mathbf{A} \leftrightarrow \mathbf{Z} \leftrightarrow (\mathbf{B}, \mathbf{W})$, and $\mathbf{Y}, \mathbf{Y}^{[t]}$ are deterministic functions of $(\mathbf{Z}, \mathbf{W})$. By the definition of ACB and Lemma 2.11 we have

$$(\mathrm{ACB}(\mathbf{A}+\mathbf{B}, \mathbf{Y}, \alpha) \approx_{2\varepsilon} \mathbf{U}_m) \mid (\{\mathrm{ACB}(\mathbf{A}+\mathbf{B}, \mathbf{Y}^i, \alpha^i\}_{i\in[t]}, \mathbf{Y}, \mathbf{Z})$$

which implies

$$(\mathrm{ACB}(\mathbf{A}+\mathbf{B}, \mathbf{Y}, \alpha) \approx_{2\varepsilon} \mathbf{U}_m) \mid (\{\mathrm{ACB}(\mathbf{A}+\mathbf{B}, \mathbf{Y}^i, \alpha^i\}_{i\in[t]}, \mathbf{W}) \tag{5.2}$$

since $\mathbf{W} = \mathbf{Y} - L(\mathbf{A}, \alpha)$ and $L(\mathbf{A}, \alpha)$ is a part of $\mathbf{Z}$. By (5.1) and (5.2) we get $\delta \leq 2\varepsilon/\gamma$. □

Next we introduce the notion of somewhere random samplers, which is motivated by the "sampler view" of the BDT error reduction we saw in Section 3.5.

Definition 5.17. $\mathrm{Samp} : \{0,1\}^n \times [D] \times [A] \to \{0,1\}^m$ *is an* (ε, δ)-somewhere random sampler *for entropy* k *if for every set* $T \subseteq \{0,1\}^m$ *s.t.* $|T| \leq \varepsilon 2^m$ *and every* (n,k)*-source* $\mathbf{X}$*,*

$$\Pr_{x\sim\mathbf{X}}\left[\Pr_{y\sim[D]}[\forall z \in [A]\ \mathrm{Samp}(x,y,z) \in T] > 2\varepsilon\right] \leq \delta.$$

We say Samp *is linear if* $\mathrm{Samp}(\cdot, y, z)$ *is linear for every* $y \in [D], z \in [A]$.

The somewhere random extractor in Theorem 4.14 can be interpreted as the following somewhere random sampler. The proof is also along the lines of Theorem 4.14.

Lemma 5.18. *For every constant* $\gamma > 0$*, and every* $\delta > 0, c < 2^{\sqrt[3]{\log(n)}}$ *there exists an explicit* $(D^{-1+\gamma}, \delta)$*-linear somewhere random sampler* $\mathrm{Samp} : \{0,1\}^n \times [D] \times [A] \to \{0,1\}^{c\log(n)}$ *for entropy* $O(c\log(n)) + \log(1/\delta)$*, where* $D = n^{O(1)}$ *and* $A = O(\log^2(c))$.

Proof. Let $\varepsilon = n^{-1/\log^2(c)}$, and fix a linear (k, ε)-seeded extractor $\mathrm{LExt} : \{0,1\}^n \times \{0,1\}^{d'} \to \{0,1\}^{c\log(n)}$ from Corollary 4.11, where $k = O(c\log(n))$. Let $\mathrm{Disp} : [D] \times [A] \to [D' = 2^{d'}]$ be the $(D^\gamma, 2\varepsilon)$-disperser from Lemma 2.21. Note that $D = 2^{2d'/\gamma} = 2^{O(d')}$ and $A = O(\frac{\log D}{\log(1/\varepsilon)}) = O(\frac{d'}{\log(1/\varepsilon)})$. Define $\mathrm{Samp}(x, s, z) := \mathrm{LExt}(x, \mathrm{Disp}(s, z))$. First note that LExt is also a (ε, δ)-sampler for min-entropy $k + \log(1/\delta)$ by Lemma 2.19, so for any source $\mathbf{X}$ with min-entropy $k + \log(1/\delta)$ and any set T of size εn^c, with probability at least $1 - \delta$ over $x \sim \mathbf{X}$ the number of y such that $\mathrm{Samp}(x, y) \in T$ is at most $2\varepsilon D'$. For any such x, let B_x be the set which consists of every s s.t. $\forall z \in [A]$, $\mathrm{LExt}(x, \mathrm{Disp}(s, z))$ falls in T. The definition of Disp implies that $|B_x| \leq D^\gamma$, which proves that Samp is a $(D^{-\gamma}, \delta)$-somewhere random sampler for entropy $k + \log(1/\delta)$. Finally, observe that $\mathrm{Samp}(\cdot, s, z)$ is linear for every s, z since LExt is linear. □

Finally we prove the following result, which will directly imply Lemma 5.15 by plugging in proper choices of somewhere random samplers and affine correlation breakers.

Lemma 5.19. *For every $\varepsilon, \delta > 0$ the following holds. Let $\mathrm{ACB} : \{0,1\}^n \times \{0,1\}^d \times [AC] \to \{0,1\}$ be a $(Ct - 1)$-strong affine correlation breaker for entropy k_1 with error $A^{-2t}C^{-1}\varepsilon\delta$, and let $\mathrm{Samp} : \{0,1\}^n \times [A] \times [C] \to \{0,1\}^d$ be a (ε, δ)-somewhere random sampler for entropy k_2. Then for every n-bit source $\mathbf{X} = \mathbf{X}_1 + \mathbf{X}_2$ such that $\mathbf{X}_1$ is independent of $\mathbf{X}_2$, $\mathrm{H}_\infty(\mathbf{X}_1) \geq k_1 + Ctd$ and $\mathrm{H}_\infty(\mathbf{X}_2) \geq k_2$, the source*

$$\mathrm{Reduce}(\mathbf{X}) := \left\{ \bigoplus_{z \in [C]} \mathrm{ACB}\left(\mathbf{X}, \mathrm{Samp}(\mathbf{X}, \alpha, z), (\alpha, z)\right) \right\}_{\alpha \in [A]}$$

is 3δ-close to a convex combination of $(2\varepsilon A, t)$-NOBF source.

Proof. Consider Lemma 5.16 by taking $\mathbf{X}_1$ as the source, $A^{-t}\delta$ as the error parameter and $L(x, (\alpha, z)) := \mathrm{Samp}(x, \alpha, z)$ as the leakage function. For every non-empty sub-

set $T \subseteq [A]$ of size at most t and every $z^* \in [C]$, define a set BAD'_{T,z^*} as follows. Let α^* denote the first element in T. Let $\beta = (\alpha^*, z^*)$ and

$$\beta' = \{(\alpha, z)\}_{\alpha \in T, z \in [C]} \backslash \{\beta\}.$$

Note that β' contains at most $2^c t - 1$ advice which are all different from β. Then we define

$$\mathsf{BAD}'_{T,z^*} := \mathsf{BAD}_{\beta,\beta'},$$

where $\mathsf{BAD}_{\beta,\beta'}$ is defined as in Lemma 5.16. Observe that by definition of BAD'_{T,z^*}, for every $x_2 \in \{0,1\}^n$, if $\mathrm{Samp}(x_2, \alpha^*, z^*) \notin \mathsf{BAD}'_{T,z^*}$, then

$$\bigoplus_{\alpha \in T} \bigoplus_{z \in [C]} \mathrm{ACB}\left(\mathbf{X}_1 + x_2, \mathrm{Samp}(\mathbf{X}_1, \alpha, z) + \mathrm{Samp}(x_2, \alpha, z), (\alpha, z)\right)$$

is $A^{-t}\delta$-close to $\mathbf{U}_1$. By the linearity of Samp, we know that for every fixing $\mathbf{X}_2 = x_2$, if $\mathrm{Samp}(x_2, \alpha^*, z^*) \notin \mathsf{BAD}'_{T,z^*}$, then

$$\left(\bigoplus_{\alpha \in T} \bigoplus_{z \in [C]} \mathrm{ACB}\left(\mathbf{X}, \mathrm{Samp}(\mathbf{X}, \alpha, z), (\alpha, z)\right)\right) \approx_{A^{-t}\delta} \mathbf{U}_1. \tag{5.3}$$

By Lemma 5.16 we know that $\Pr_{y \sim \mathbf{U}_d}\left[y \in \mathsf{BAD}'_{T,z^*}\right] \leq A^{-t}C^{-1}\varepsilon$. Now define BAD' to be the union of BAD'_{T,z^*} for all possible choices of T, z^*. Since there are at most A^t choices of T and C choices of z^*, by union bound we know that $\Pr_{y \sim \mathbf{U}_d}[y \in \mathsf{BAD}'] \leq \varepsilon$. Therefore, by definition of somewhere random sampler,

$$\Pr_{x_2 \sim \mathbf{X}_2}\left[|\{\alpha \in [A] : \forall z\ \mathrm{Samp}(x_2, \alpha, z) \in \mathsf{BAD}'\}| \leq 2\varepsilon A\right] \geq 1 - \delta.$$

In other words, with probability at least $1 - \delta$ over the fixing $\mathbf{X}_2 = x_2$, there exists a set $Q \subseteq [A]$ of size at most $2\varepsilon A$ which satisfies the following: for every $\alpha \in [A] \backslash Q$, there exists z_α such that $\mathrm{Samp}(x_2, \alpha, z_\alpha) \notin \mathsf{BAD}'$, which also implies $\mathrm{Samp}(x_2, \alpha, z_\alpha) \notin \mathrm{BAD}'_{T,z_\alpha}$. By Equation (5.3), for every $T \subseteq [A] \backslash Q$ s.t. $1 \leq |T| \leq t$,

$$\left(\bigoplus_{\alpha \in T} \bigoplus_{z \in \{0,1\}^c} \mathrm{ACB}(\mathbf{X}, \mathrm{Samp}(\mathbf{X}, \alpha, z), (\alpha, z))\right) \approx_{A^{-t}\delta} \mathbf{U}_1.$$

By Lemma 2.4 this implies that with probability $1-\delta$ over the fixing of $\mathbf{X}_2$,

$$\mathrm{Reduce}(\mathbf{X}) = \left\{ \bigoplus_{z\in\{0,1\}^c} \mathrm{ACB}(\mathbf{X}, \mathrm{Samp}(\mathbf{X}, \alpha, z), (\alpha, z)) \right\}_{\alpha\in[A]}$$

is 2δ-close to a $(2\varepsilon A, t)$-NOBF source. Therefore $\mathrm{Reduce}(\mathbf{X})$ is 3δ-close to a convex combination of $(2\varepsilon A, t)$-NOBF source. □

Now we are ready to prove Lemma 5.15.

Proof of Lemma 5.15. Let $\mathrm{Samp} : \{0,1\}^n \times [N] \times [C] \to \{0,1\}^d$ be a $\left(\frac{N^{-\gamma}}{2}, \frac{N^{-\gamma}}{3}\right)$-somewhere random sampler from *Lemma* 5.18, where $N = n^{O(1)}$. We want to choose proper parameters d, C so that there exists a $(Ct-1)$-strong affine correlation breaker $\mathrm{ACB} : \{0,1\}^n \times \{0,1\}^d \times [NC] \to \{0,1\}$ with error $N^{-2(t+\gamma)}C^{-1}/6$. Then Lemma 5.19 would imply Lemma 5.15. Observe that we need to guarantee

$$d \geq K_1 \left(Ct^2 \log(n) \cdot \left(\frac{\log\log(n)}{\log\log\log(n)} + \log^3(Ct) \right) \right)$$

and

$$C \geq K_2 \log^2 \left(\frac{d}{\log(n)} \right)$$

for some fixed constants K_1, K_2. It suffices to take

$$C = O(\log\log\log^2(n) + \log^2(t))$$

for some large enough constant factor. Then the entropy requirement of ACB would be

$$k_1 = O\left(Ct^2 \log(n) \cdot \left(\frac{\log\log(n)}{\log\log\log(n)} + Ct \right) \right),$$

and the entropy requirement of Samp would be

$$k_2 = O(d + \log(N^\gamma)) = O(d + \log(n)).$$

To make Reduce work, the entropy of the given sumset source should be at least

$$\begin{aligned} k &= \max(k_1 + Ctd, k_2) \\ &= O\left(C^2 t^3 \log(n) \cdot \left(\frac{\log\log(n)}{\log\log\log(n)} + \log^3(t)\right)\right). \end{aligned}$$

Finally, to prove the explicitness of Reduce, observe that the running time of Reduce is N times the running time of ACB and Samp, which is also $\mathrm{poly}(n)$. □

5.4 Improved Reduction for Small-space Sources

Our improved small-space extractor results are based on the following key lemma.

Lemma 5.20. *For every integer $C \geq 2$, every space-s source on n-bit with min-entropy*

$$k' \geq Ck + (C-1)(2s + 2\log(n/\varepsilon))$$

is $(3C\varepsilon)$-close to a convex combination of C-sumset sources with min-entropy k.

Note that by taking $C = 2$ in Lemma 5.20, we get that the sumset source extractor in Theorem 5.2 and Theorem 5.3 are also small-space source extractors which satisfy the parameters in Theorem 5.8 and Theorem 5.9 respectively. In the rest of this section we focus on proving Lemma 5.20. First we show how to derive Lemma 5.20 based on the following lemma.

Lemma 5.21. *Every space-s source $\mathbf{X} \in \{0,1\}^n$ with entropy at least $k = k_1 + k_2 + 2s + 2\log(n/\varepsilon)$ is 3ε-close to a convex combination of sources of the form $\mathbf{X}_1 \circ \mathbf{X}_2$ which satisfy the following properties:*

- *$\mathbf{X}_1$ is independent of $\mathbf{X}_2$*

- $\mathrm{H}_\infty(\mathbf{X}_1) \geq k_1$, $\mathrm{H}_\infty(\mathbf{X}_2) \geq k_2$
- $\mathbf{X}_2$ *is a space-*s *source*

Proof of Lemma 5.20. By induction, Lemma 5.21 implies that a space-s source with entropy $Ck+(C-1)(2s+2\log(n/\varepsilon))$ is $3C\varepsilon$-close to a convex combination of sources of the form $\mathbf{X}_1 \circ \mathbf{X}_2 \circ \cdots \circ \mathbf{X}_C$ where $\mathbf{X}_1, \ldots, \mathbf{X}_C$ are independent, and for every $i \in [C]$, $\mathrm{H}_\infty(\mathbf{X}_i) \geq k$. Let $\ell_1, \ell_2, \ldots, \ell_C$ denote the length of $\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_C$ respectively and define $p_i = \sum_{j=1}^{i-1} \ell_i$ and $s_i = \sum_{j=i+1}^{n} \ell_j$ (note that $p_1 = 0$ and $s_C = 0$). Then observe that

$$\mathbf{X}_1 \circ \cdots \circ \mathbf{X}_C = \sum_{i=1}^{C} 0^{p_i} \circ \mathbf{X}_i \circ 0^{s_i},$$

which implies that $\mathbf{X} = \mathbf{X}_1 \circ \cdots \circ \mathbf{X}_C$ is a C-sumset source with min-entropy k. □

Next we prove Lemma 5.21. To make the discussion easier, first we define a branching program which is a layered graph that captures the sampling process in Definition 5.7 in the sense that each layer corresponds to a time step and each vertex in a layer corresponds to a state in a certain time step.

Definition 5.22. *A branching program B of width w and length n (for sampling) is a directed (multi)-graph with $(n+1)$ layers $L_0, L_1, \ldots, L_n$ and has at most w vertices in each layer. The first layer (indexed by 0) has only one vertex called the start vertex, and every vertex in L_n has no outgoing edge. For every vertex v in layer $i < n$, the set of outgoing edges from v, denoted by E_v, satisfies the following.*

- *Every edge $e \in E_v$ is connected to a vertex in L_{i+1}.*
- *Each edge $e \in E_v$ is labeled with a probability, denoted by $\Pr[e]$, so that $\sum_{e \in E_v} \Pr[e] = 1$.*

- *Each edge $e \in E_v$ is labeled with a bit $b_e \in \{0,1\}$, and if two distinct edges $e_1, e_2 \in E_v$ are connected to the same vertex $w \in L_{i+1}$ then $b_{e_1} \neq b_{e_2}$. (Note that this implies $|E_v| \leq 2w$.)*

The output of B is a n-bit string generated by the following process. Let v_0 be the start vertex. Repeat the following for i from 1 to n: sample an edge $e_i \in E_{v_{i-1}}$ with probability $\Pr[e_i]$, output b_{e_i} and let v_i be the vertex which is connected by e_i. We say $(v_0, e_1, v_1, \ldots, e_n, v_n)$ is the computation path *of B. We say a random variable $\mathbf{X} \in \{0,1\}^n$ is a space-s source if it is generated by a branching program of width 2^s and length n.*

We also consider the subprograms of a branching program.

Definition 5.23. *Let $B = (L_0, L_1, \ldots, L_n)$ be a branching program of width w and length n and let v be a vertex in layer i of B. Then the subprogram of B starting at v, denoted by B_v, is the induced subgraph of B which consists of $(\{v\}, L_{i+1}, \ldots, L_n)$. Note that B_v is a branching program of width w and length $n - i$ which takes v as the start vertex.*

The following simple fact is from [KRVZ11].

Lemma 5.24 ([KRVZ11])**.** *Let $\mathbf{X}$ be a space-s source sampled by a branching program B, and let v be a vertex in layer i of B. Then conditioned on the event that the computation path of $\mathbf{X}$ passes v, $\mathbf{X}$ is the concatenation of two independent random variables $\mathbf{X}_1 \in \{0,1\}^i$, $\mathbf{X}_2 \in \{0,1\}^{n-i}$. Moreover $\mathbf{X}_2$ is exactly the source generated by the subprogram B_v.*

We also need the following lemma.

Lemma 5.25. *Let B be a branching program of width 2^s and length n for sampling. Let e be an edge in B connected from u to v and let $\mathbf{X}_u, \mathbf{X}_v$ be the output distributions of the subprograms B_u, B_v respectively. Then $\mathrm{H}_\infty(\mathbf{X}_v) \geq \mathrm{H}_\infty(\mathbf{X}_u) - \log(1/\Pr[e])$.*

Proof. Let $x^* = \arg\max_x \Pr[\mathbf{X}_v = x]$. Note that

$$\mathrm{H}_\infty(\mathbf{X}_v) = -\log(\Pr[\mathbf{X}_v = x^*])$$

by definition. Observe that

$$\Pr[\mathbf{X}_u = b_e \circ x^*] \geq \Pr[e] \cdot \Pr[\mathbf{X}_v = x^*].$$

Therefore,

$$\begin{aligned}
\mathrm{H}_\infty(\mathbf{X}_u) &\leq -\log\left(\Pr[\mathbf{X}_u = b_e \circ x^*]\right) \\
&\leq -\log\left(\Pr[e] \cdot \Pr[\mathbf{X}_v = x^*]\right) \\
&= \mathrm{H}_\infty(\mathbf{X}_v) + \log(1/\Pr[e]).
\end{aligned}$$

□

Now we are ready to prove Lemma 5.21.

Proof of Lemma 5.21. Let B denote the branching program that samples $\mathbf{X}$. For every v, define $\mathbf{X}_v$ to be the source generated by the subprogram B_v. Define v to be a *stopping vertex* if

$$\mathrm{H}_\infty(\mathbf{X}_v) \leq k_2 + s + \log(n/\varepsilon).$$

Observe that every vertex u in the last layer is a stopping vertex since $\mathrm{H}_\infty(\mathbf{X}_u) = 0$. Therefore there is always a stopping vertex in the computation path. We define an edge e in B to be a *bad edge* if

$$\Pr[e] \leq \varepsilon/(n \cdot 2^s).$$

Now define a random variable $\mathbf{V}$ as follows:

- $\mathbf{V} = \perp$ if the computation path of $\mathbf{X}$ visits a bad edge before visiting any stopping vertex,

- otherwise, $\mathbf{V} = v$ where v is the first stopping vertex in the computation path.

Observe that $\Pr[\mathbf{V} = \perp] \leq 2\varepsilon$, since in each step of B there are at most 2^{s+1} edges starting from the current vertex, and there are n steps in total. Define

$$\mathsf{BAD} = \{v \in \mathrm{Supp}(\mathbf{V}) : \mathrm{H}_\infty(\mathbf{X}|_{\mathbf{V}=v}) \leq k - s - \log(n/\varepsilon)\}.$$

Then $\Pr[\mathbf{V} \in \mathsf{BAD}] \leq \varepsilon$ by Lemma 2.6. We claim that if $v \notin \mathsf{BAD}$ and $v \neq \perp$, then conditioned on $\mathbf{V} = v$, the source $\mathbf{X}$ can be written as $\mathbf{X}_1 \circ \mathbf{X}_2$ which satisfies the properties stated in Lemma 5.21. The claim directly implies Lemma 5.21 because $\Pr[v \in \mathsf{BAD} \vee v = \perp] \leq 3\varepsilon$ by union bound. Next we prove the claim. Let E_1 denote the event "the computation path contains v", and E_2 denote the event "the computation path does not contain any bad edge or stopping vertex before the layer of v". Observe that $\mathbf{V} = v$ is equivalent to $E_1 \wedge E_2$. Conditioned on E_1, by Lemma 5.24, $\mathbf{X}$ can be written as $\mathbf{X}_1 \circ \mathbf{X}_2$ where $\mathbf{X}_1$ is independent of $\mathbf{X}_2$ and $\mathbf{X}_2 = \mathbf{X}_v$. Now observe that E_2 only involves layers before v, so conditioned on E_1, $\mathbf{X}_2$ is independent of E_2. Therefore, conditioned on $\mathbf{V} = v$, we still have $\mathbf{X}_2 = \mathbf{X}_v$, which is a space-$s$ source, and $\mathbf{X}_1$ is still independent of $\mathbf{X}_2$. Next observe that

$$\begin{aligned}
\mathrm{H}_\infty(\mathbf{X}_1) &= \mathrm{H}_\infty(\mathbf{X}|_{\mathbf{V}=v}) - \mathrm{H}_\infty(\mathbf{X}_2) \\
&\geq (k - s - \log(n/\varepsilon)) - (k_2 + s + \log(n/\varepsilon)) \\
&\geq k_1.
\end{aligned}$$

It remains to prove that $\mathrm{H}_\infty(\mathbf{X}_2) \geq k_2$. Assume for contradiction that $\mathrm{H}_\infty(\mathbf{X}_v) < k_2$. Let e be the edge in the computation path which connects to v, and suppose e is from u. Now consider the following two cases.

- If e is not a bad edge, then

$$\mathrm{H}_\infty(\mathbf{X}_u) \leq \mathrm{H}_\infty(\mathbf{X}_v) + \log(1/\Pr[e]) < k_2 + s + \log(n/\varepsilon),$$

which means u is also a stopping vertex. Therefore v cannot be the first stopping vertex.

- If e is a bad edge, then either there is a stopping vertex before e or $\mathbf{V} = \perp$.

In both cases $\mathbf{V} \neq v$, which is a contradiction. In conclusion we must have $\mathrm{H}_\infty(\mathbf{X}_2) \geq k_2$. □

5.5 Inner Product as a Sumset Extractor

In this section we prove that inner product function is a sumset extractor. First we show that the following lemma implies Theorem 5.10.

Lemma 5.26. *Let $H : \{0,1\}^n \to \{0,1\}$ be a function s.t. $\Pr_{x\sim\mathbf{U}_n}[H(x+y) = H(x)] = 1/2$ for any non-zero $y \in \{0,1\}^n$. Then H is an extractor for (n, k_1, k_2)-sumset source with error ε, where $\varepsilon = 2^{\frac{n-(k_1+k_2)}{2}-1}$.*

Proof of Theorem 5.10. It suffices to show that $\Pr_{x\sim\mathbf{U}_n}[\mathsf{IP}(x+y) = \mathsf{IP}(x)] = 1/2$ for every non-zero y. First we split x and y into halves so that $x = (x_1, x_2)$ and $y = (y_1, y_2)$. Then observe that

$$\mathsf{IP}(x+y) + \mathsf{IP}(x) = \langle x_1, y_2\rangle + \langle y_1, x_2\rangle + \langle y_1, y_2\rangle .^{13}$$

In addition, since y is non-zero, either y_1 or y_2 is non-zero. If y_1 is non-zero, then,

$$\Pr_{x\sim\mathbf{U}_n}[\mathsf{IP}(x+y) = \mathsf{IP}(x)] = \Pr_{x_1\sim\mathbf{U}_{n/2}}[\langle x_1, y_1\rangle = \langle y_1, y_2\rangle + \langle y_1, x_2\rangle] = 1/2,$$

using the fact that $\langle y_1, \mathbf{U}_{n/2}\rangle$ is uniform for any non-zero y_1. The case that y_2 is non-zero is similar. □

[13] $\langle\cdot,\cdot\rangle$ denotes the inner product of vectors over $\mathbb{F}_2^{n/2}$.

Proof of Lemma 5.26.

$$\Delta\left(H(\mathbf{X}+\mathbf{Y});\mathbf{U}_1\right) = \frac{1}{2}\mathop{\mathbb{E}}_{x\sim\mathbf{X},y\sim\mathbf{Y}}\left[(-1)^{H(x+y)}\right]$$

$$\leq \frac{1}{2}\sqrt{\mathop{\mathbb{E}}_{x\sim\mathbf{X}}\left[\left(\mathop{\mathbb{E}}_{y\sim\mathbf{Y}}\left[(-1)^{H(x+y)}\right]\right)^2\right]}$$

(Cauchy-Schwarz)

$$\leq \frac{1}{2}\sqrt{\sum_{x\in\{0,1\}^n} 2^{-k_1}\left(\mathop{\mathbb{E}}_{y\sim\mathbf{Y}}\left[(-1)^{H(x+y)}\right]\right)^2}$$

($\mathrm{H}_\infty(\mathbf{X}) \geq k_1$)

$$= \frac{1}{2}\sqrt{2^{n-k_1}\mathop{\mathbb{E}}_{y_1\sim\mathbf{Y},y_2\sim\mathbf{Y}}\left[\mathop{\mathbb{E}}_{x\sim\mathbf{U}}\left[(-1)^{H(x+y_1)+H(x+y_2)}\right]\right]}$$

$$= \frac{1}{2}\sqrt{2^{n-k_1}\mathop{\mathbb{E}}_{y_1\sim\mathbf{Y},y_2\sim\mathbf{Y}}\left[\mathop{\mathbb{E}}_{x\sim\mathbf{U}}\left[(-1)^{H(x)+H(x+(y_2-y_1))}\right]\right]}$$

$$= \frac{1}{2}\sqrt{2^{n-k_1}\cdot\Pr_{y_1\sim\mathbf{Y},y_2\sim\mathbf{Y}}[y_1=y_2]}$$

(inner formula is 1 if $y_1 = y_2$ and 0 if not)

$$= 2^{\frac{n-(k_1+k_2)}{2}-1}.$$

($\mathrm{H}_\infty(\mathbf{Y}) \geq k_2$)

□

5.6 Small-doubling Sumset Sources

In this section we prove that a sumset source $\mathbf{A}+\mathbf{B}$ with small doubling constant is close to an affine source. First we consider the case that the supports of $\mathbf{A}$ and $\mathbf{B}$ are dense sets. We note that it standard in additive combinatorics to reduce the small-doubling setting to the dense-set setting by Freiman homomorphism [Fre73]. Our proof in this case relies on the following "invariant lemma" by Sanders.

Lemma 5.27. *Consider $A, B \subseteq \mathbb{F}_2^m$ such that $|A|, |B| \geq |\mathbb{F}_2^m|/r$, and let $\mathbf{A}, \mathbf{B}$ be uniform distributions over A, B respectively. Then for any function $f : \mathbb{F}_2^m \to [0,1]$, there exists a*

distribution $\mathbf{T} \subseteq \mathbb{F}_2^m$ *and a linear subspace* V *of co-dimension* $O(\log^4(r))$ *such that*

$$\mathbb{E}\left[f(\mathbf{A}+\mathbf{B})\right] \approx_\varepsilon \mathbb{E}\left[f(\mathbf{T}+\mathbf{V})\right],$$

where $\mathbf{V}$ *is the uniform distribution over* V*.*

Now we see that for any distinguisher f, there is a convex combination of affine sources $\mathbf{T}+\mathbf{V}$ that is indistinguishable by f. Therefore we can use von Neuman's minimax theorem to prove that there is a convex combination of affine sources that is indistinguishable from $\mathbf{A}+\mathbf{B}$ by every distinguisher. It remains to show that the same argument also works for pair of sets $(A, B) \subseteq (\mathbb{F}_2^n)^2$ with small doubling constant r. Suppose $\mathbf{A}$ and $\mathbf{B}$ are random variables that are uniform over A and B respectively. We want to show that for any function $f : \mathbb{F}_2^n \to [0,1]$ there is also a convex combination of affine sources that is close to $\mathbf{A}+\mathbf{B}$. To reduce to the setting of dense sets, we need the following two lemmas. The first lemma is a standard application of Freiman homomorphism.

Lemma 5.28. *For any pair of sets* $A, B \subseteq \mathbb{F}_2^n$ *with the same size that has doubling constant* r*, there exists a linear mapping* $\phi : \mathbb{F}_2^n \to \mathbb{F}_2^m$ *and its "inverse"* $\phi^{-1} : \mathbb{F}_2^m \to \mathbb{F}_2^n$ *which satisfy the following property.*

- $2^m < \mathrm{poly}(r)\,|A|$
- ϕ *is injective on* A*,* B *and* $A+B$*,*
- $\phi^{-1}(\phi(x)) = x$ *for every* $x \in A+B$
- *For any affine subspace* $V \subseteq \mathbb{F}_2^m$ *that satisfies* $|V \cap \phi(A+B)| > 1/2\,|V|$*,* ϕ^{-1} *is injective on* V*, and* $\phi^{-1}(V)$ *is also an affine subspace.*

The second lemma is a generalized version of Sanders' invariance lemma, which shows that we can find a distribution $\mathbf{T}+\mathbf{V}$ that is indistinguishable from $\mathbf{A}+\mathbf{B}$ by two functions simultaneously.

Lemma 5.29. *Let $A, B \subseteq \mathbb{F}_2^n$ be sets which satisfy $|A|, |B| \geq |\mathbb{F}_2^n|/r$. Let $\mathbf{A}, \mathbf{B}$ be the uniform distributions over A, B respectively. Then for every $\varepsilon > 0$ and every pair of functions $f, g : \mathbb{F}_2^n \to [0,1]$ there exists a linear subspace V of co-dimension $O(\log^3(r/\varepsilon)\log(r)/\varepsilon^2)$ and a distribution $\mathbf{T} \in \mathbb{F}_2^n$ such that*

$$\mathbb{E}[f(\mathbf{A}+\mathbf{B})] \approx_\varepsilon \mathbb{E}[f(\mathbf{T}+\mathbf{V})]$$

and

$$\mathbb{E}[g(\mathbf{A}+\mathbf{B})] \approx_\varepsilon \mathbb{E}[g(\mathbf{T}+\mathbf{V})],$$

where $\mathbf{V}$ is the uniform distribution over V.

In addition, the minimax argument that we mentioned above can be formalized as follows.

Lemma 5.30. *Let Ω be a finite set, $\mathcal{X}$ be a convex set of distributions on Ω, and $\mathbf{Y}$ be a distribution on Ω. If for every function $f : \Omega \to [0,1]$ there exists $\mathbf{X}_f \in \mathcal{X}$ such that $\mathbb{E}[f(\mathbf{X}_f)] - \mathbb{E}[f(\mathbf{Y})] \leq \varepsilon$, then there exists $\mathbf{X}^* \in \mathcal{X}$ such that $\mathbf{Y} \approx_\varepsilon \mathbf{X}^*$.*

Before we prove these lemmas, first we prove Theorem 5.13 assuming the lemmas above are correct.

Proof of Theorem 5.13. Consider any function $f : \mathbb{F}_2^n \to [0,1]$. Let $\phi : \mathbb{F}_2^n \to \mathbb{F}_2^m$ and $\phi^{-1} : \mathbb{F}_2^n \to \mathbb{F}_2^m$ be the mappsing in Lemma 5.28. Let $A' = \phi(A), B' = \phi(B), \mathbf{A}' = \phi(\mathbf{A}), \mathbf{B}' = \phi(\mathbf{B})$. Because ϕ is injective on A and B, $\mathbf{A}', \mathbf{B}'$ are exactly the uniform distributions over A', B' respectively. In addition, $|A'| = |B'| = |A| \geq |\mathbb{F}_2^m|/r^{13}$. By Lemma 5.29, there exists a distribution $\mathbf{T} \in \mathbb{F}_2^m$ and a linear subspace V of entropy $k' = m - O(\log(r)\log(r/\varepsilon)^3/\varepsilon^2)$ such that

$$\mathbb{E}[\mathbb{1}_{A'+B'}(\mathbf{A}'+\mathbf{B}')] \approx_{\varepsilon/3} \mathbb{E}[\mathbb{1}_{A'+B'}(\mathbf{T}+\mathbf{V})]$$

and

$$\mathbb{E}\left[f(\phi^{-1}(\mathbf{A}'+\mathbf{B}'))\right] \approx_{\varepsilon/3} \mathbb{E}\left[f(\phi^{-1}(\mathbf{T}+\mathbf{V}))\right], \tag{5.4}$$

where $\mathbf{V}$ is the uniform distribution over V. Now observe that since $\mathbb{E}\left[\mathbb{1}_{A'+B'}(\mathbf{A}'+\mathbf{B}')\right] = 1$,

$$\mathbb{E}\left[\mathbb{1}_{A'+B'}(\mathbf{T}+\mathbf{V})\right] \geq 1-\varepsilon/3.$$

By Markov's inequality,

$$\Pr_{t\sim\mathbf{T}}\left[\mathbb{E}\left[\mathbb{1}_{A'+B'}(t+\mathbf{V})\right] > 1/2\right] \geq 1-2\varepsilon/3.$$

In other words,

$$\Pr_{t\sim\mathbf{T}}\left[|\phi(A+B)\cap(t+V)| > \frac{1}{2}|t+V|\right] \geq 1-2\varepsilon/3.$$

By Lemma 5.28,

$$\Pr_{t\sim\mathbf{T}}\left[\phi^{-1}(t+\mathbf{V}) \text{ is an affine source of entropy } k'\right] \geq 1-2\varepsilon/3.$$

Therefore $\phi^{-1}(\mathbf{T}+\mathbf{V})$ is $(2\varepsilon/3)$-close to a convex combination of affine sources (denoted by $\mathbf{W}$) of entropy k'. Finally, note that $\phi^{-1}(\mathbf{A}'+\mathbf{B}') = \phi^{-1}(\phi(\mathbf{A}+\mathbf{B}))$ is exactly $\mathbf{A}+\mathbf{B}$. Therefore by (5.4) and triangle inequality,

$$\mathbb{E}\left[f(\mathbf{A}+\mathbf{B})\right] \approx_{\varepsilon} \mathbb{E}\left[f(\mathbf{W})\right].$$

Since the proof above works for every function $f:\mathbb{F}_2^n\to[0,1]$, by Lemma 5.30, $\mathbf{A}+\mathbf{B}$ is ε-close to a convex combination of affine sources. □

Next we prove the lemmas one by one.

5.6.1 Freiman Homomorphism

In this section we prove Lemma 5.28. First we define the Freiman homomorphism. Here we restrict our attention to the linear case, so that the definition can be simplified as follows.

Definition 5.31. *We say a function $\phi : \mathbb{F}_2^n \to \mathbb{F}_2^m$ is a linear s-Freiman homomorphism of a set $A \subseteq \mathbb{F}_2^n$ if ϕ is linear and injective on sA.*

The following property is easy to verify.

Lemma 5.32. *If ϕ is a linear s-Freiman homomorphism, then ϕ is injective on $sA + v$ for every $v \in \mathbb{F}_2^n$. Further, for $x \in 2sA$ we have $\phi(x) = 0 \Leftrightarrow x = 0$.*

A result by Green and Ruzsa [GR07] shows the existence of a linear s-Freiman homomorphism for any set A with codomain size at most $|2sA|$.

Lemma 5.33 ([GR07])**.** *For every set $A \subseteq \mathbb{F}_2^n$, there exists a linear s-Freiman homomorphism $\phi : \mathbb{F}_2^n \to \mathbb{F}_2^m$ of A such that $\phi(2sA) = \mathbb{F}_2^m$.*

This can be combined with the the Plunnecke-Ruzsa lemma to bound the size of 2^m.

Lemma 5.34 ([Plü61, Ruz99])**.** *For every $A, B \subseteq \mathbb{F}_2^n$ s.t. $|A| = |B|$ and $|A + B| \leq r\,|A|$, $|kA + \ell B| \leq r^{k+\ell+1}\,|A|$ for every $k, \ell \in \mathbb{N}$.*

Now we are ready to prove Lemma 5.28.

Proof of Lemma 5.28. Let $\phi : \mathbb{F}_2^n \to \mathbb{F}_2^m$ be the 3-Freiman homomorphism of $A + B$, with size at most $|6A + 6B|$ (Lemma 5.33). By Lemma 5.34 we have $2^m \leq r^1 3\,|A|$, i.e., the first condition holds. The second condition is trivial by Lemma 5.32. To

prove the other conditions, take ϕ^{-1} to be any mapping that satisfies $\phi^{-1}(\phi(x))$ for every $x \in 3A+3B$. The third condition holds because $A+B \subseteq 3A+3B$. To prove the last condition, let $X = A + B$, and let t be an element in X such that $\phi(t) \in V$. Note that t must exist because $V \cap \phi(X)$ is non-empty. Since V is an affine subspace, for every $v \in V$ and $v_1 \in V \cap \phi(X)$, $v + \phi(t) - v_1 \in V$. Because $|V \cap \phi(X)| > |V \backslash \phi(X)|$, for every $v \in V$ there must exist $v_1 \in V \cap \phi(X)$ s.t. $v + \phi(t) - v_1 \in V \cap \phi(X)$. In other words, for every $v \in V$ there exist $v_1, v_2 \in V \cap \phi(X)$ such that $v = v_1 + v_2 - \phi(t)$. This means $V \subseteq \phi(2X - t) \subseteq \phi(3X)$. Because ϕ^{-1} is injective on $\phi(3X)$, this implies that ϕ^{-1} is injective on V. Next we prove that $\phi^{-1}(V)$ is also an affine subspace. It suffices to prove that for every $u, v \in V$,

$$\phi^{-1}(u) + \phi^{-1}(v) - t = \phi^{-1}(u + v - \phi(t)),$$

because $\phi^{-1}(u + v - \phi(t)) \in \phi^{-1}(V)$. Observe that

$$\phi(\phi^{-1}(u) + \phi^{-1}(v) - t - \phi^{-1}(u + v - \phi(t))) = u + v - \phi(t) - (u + v - \phi(t)) = 0,$$

because ϕ is linear, and for every $y \in \{u, v, u + v - \phi(t)\}$ we have $y \in V \subseteq \phi(3X)$, which means $\phi(\phi^{-1}(y)) = y$. Moreover, because $\phi^{-1}(u), \phi^{-1}(v), \phi^{-1}(u + v - \phi(t)) \in \phi^{-1}(V) \subseteq 2X - t$,

$$\phi^{-1}(u) + \phi^{-1}(v) - t - \phi^{-1}(u + v - \phi(t)) \in 6X.$$

By Lemma 5.32, $\phi^{-1}(u) + \phi^{-1}(v) - t - \phi^{-1}(u + v - \phi(t)) = 0$. □

5.6.2 Generalized Invariant Lemma

In this section we prove Lemma 5.29. The proof is along the lines of [San12, Theorem A.1]. (See also the survey by Lovett [Lov15].) The only difference is we generalize the Croot-Sisask lemma [CS10] used in the proof to simultaneously work for two

functions. We formalize the generalized Croot-Sisask lemma below, and we prove it following an exposition by Ben-Sasson, Ron-Zewi, Tulsiani and Wolf [BRTW14] which is more convenient for our setting.

Lemma 5.35. *Let $A \subseteq \mathbb{F}_2^n$ be a set which satisfies $|A| \geq |\mathbb{F}_2^n|/r$. Then for every $\varepsilon > 0$ and every pair of functions $f, g : \mathbb{F}_2^n \to [0,1]$ there exists $t = O(\log(r/\varepsilon)/\varepsilon^2)$ and a set X of size at least $|\mathbb{F}_2^n|/2r^t$ such that for every set B s.t. $|B| \geq |\mathbb{F}_2^n|/r$ and every $x \in X$,*

$$\mathop{\mathbb{E}}_{a\sim A, b\sim B}[f(a+b)] \approx_\varepsilon \mathop{\mathbb{E}}_{a\sim A, b\sim B}[f(a+b+x)]$$

and

$$\mathop{\mathbb{E}}_{a\sim A, b\sim B}[g(a+b)] \approx_\varepsilon \mathop{\mathbb{E}}_{a\sim A, b\sim B}[g(a+b+x)].$$

Proof. Let $t = 8\ln(128r/\varepsilon)/\varepsilon^2$. By Chernoff-Hoeffding bound, for every $b \in \mathbb{F}_2^n$,

$$\Pr_{(a_1,\ldots,a_t)\sim A^t}\left[\frac{1}{t}\sum_{i=1}^t f(a_i+b) \approx_{\frac{\varepsilon}{4}} \mathop{\mathbb{E}}_{a\sim A}[f(a+b)]\right] \geq 1 - \frac{\varepsilon}{16r}$$

and

$$\Pr_{(a_1,\ldots,a_t)\sim A^t}\left[\frac{1}{t}\sum_{i=1}^t g(a_i+b) \approx_{\frac{\varepsilon}{4}} \mathop{\mathbb{E}}_{a\sim A}[g(a+b)]\right] \geq 1 - \frac{\varepsilon}{16r}.$$

Then by union bound and by averaging over $b \sim \mathbb{F}_2^n$,

$$\Pr_{\substack{(a_1,\ldots,a_t)\sim A^t \\ b\sim\mathbb{F}_2^n}}\left[\frac{1}{t}\sum_{i=1}^t f(a_i+b) \approx_{\frac{\varepsilon}{4}} \mathop{\mathbb{E}}_{a\sim A}[f(a+b)] \text{ and } \frac{1}{t}\sum_{i=1}^t g(a_i+b) \approx_{\frac{\varepsilon}{4}} \mathop{\mathbb{E}}_{a\sim A}[g(a+b)]\right] \geq 1 - \frac{\varepsilon}{8r}.$$

Define

$$\mathsf{BAD}_{(a_1,\ldots,a_t)} := \left\{b : \frac{1}{t}\sum_{i=1}^t f(a_i+b) \not\approx_{\frac{\varepsilon}{4}} \mathop{\mathbb{E}}_{a\sim A}[f(a+b)] \text{ or } \frac{1}{t}\sum_{i=1}^t g(a_i+b) \not\approx_{\frac{\varepsilon}{4}} \mathop{\mathbb{E}}_{a\sim A}[g(a+b)]\right\}.$$

By Markov inequality, there exists $S \subseteq A^t$ such that $|S| \geq |A|^t/2$ and for every $(a_1,\ldots,a_t) \in S$,

$$\left|\mathsf{BAD}_{(a_1,\ldots,a_t)}\right| \leq \frac{\varepsilon}{4r}|\mathbb{F}_2^n|.$$

Now classify the elements in S by $(a_2-a_1, a_3-a_1, \ldots, a_t-a_1)$. By averaging there exists a subset $X' \subseteq S$ and a $(t-1)$-tuple $(y_2, \ldots, y_t)$ such that $|X'| \geq |S| / |\mathbb{F}_2^n|^{t-1} \geq |\mathbb{F}_2^n| / 2r^t$, and for every $(a_1, \ldots, a_t) \in X'$ we have $a_i - a_1 = y_i$ for every $2 \leq i \leq t$. Let $(a_1^*, \ldots, a_t^*)$ be an element in X'. Observe that for every $(a_1, \ldots, a_t) \in X'$, $a_1 - a_1^* = \cdots = a_t - a_t^*$. Define

$$X = \{x = a_1 - a_1^* : (a_1, \ldots, a_t) \in X'\}.$$

Note that $|X| = |X'| \geq |\mathbb{F}_2^n| / 2r^t$. It remains to prove that for every $x \in X$,

$$\mathop{\mathbb{E}}_{a\sim A, b\sim B}[f(a+b)] \approx_\varepsilon \mathop{\mathbb{E}}_{a\sim A, b\sim B}[f(a+b+x)]$$

and

$$\mathop{\mathbb{E}}_{a\sim A, b\sim B}[g(a+b)] \approx_\varepsilon \mathop{\mathbb{E}}_{a\sim A, b\sim B}[g(a+b+x)].$$

Let $(a_1, \ldots, a_t) = (a_1^* + x, \ldots, a_t^* + x)$. Since $(a_1, \ldots, a_t)$ is an element in S,

$$\left| \mathop{\mathbb{E}}_{a\sim A, b\sim B}[f(a+b)] - \mathop{\mathbb{E}}_{b\sim B}\left[\frac{1}{t}\sum_{i=1}^{t} f(a_i+b)\right] \right| \leq \frac{\varepsilon}{4} + \Pr_{b\sim B}\left[b \in \mathsf{BAD}_{(a_1,\ldots,a_t)}\right] \leq \frac{\varepsilon}{2}.$$

Similarly, since $(a_1^*, \ldots, a_t^*)$ is an element in S,

$$\left| \mathop{\mathbb{E}}_{a\sim A, b\sim B}[f(a+b+x)] - \mathop{\mathbb{E}}_{b\sim B}\left[\frac{1}{t}\sum_{i=1}^{t} f(a_i^*+b+x)\right] \right| \leq \frac{\varepsilon}{4} + \Pr_{b\sim B}\left[(b+x) \in \mathsf{BAD}_{(a_1^*,\ldots,a_t^*)}\right] \leq \frac{\varepsilon}{2}.$$

Finally, observe that

$$\mathop{\mathbb{E}}_{b\sim B}\left[\frac{1}{t}\sum_{i=1}^{t} f(a_i+b)\right] = \mathop{\mathbb{E}}_{b\sim B}\left[\frac{1}{t}\sum_{i=1}^{t} f(a_i^*+x+b)\right].$$

By triangle inequality we can conclude that

$$\mathop{\mathbb{E}}_{a\sim A, b\sim B}[f(a+b)] \approx_\varepsilon \mathop{\mathbb{E}}_{a\sim A, b\sim B}[f(a+b+x)].$$

Similarly we can prove that

$$\mathop{\mathbb{E}}_{a\sim A, b\sim B}[g(a+b)] \approx_\varepsilon \mathop{\mathbb{E}}_{a\sim A, b\sim B}[g(a+b+x)].$$

□

In addition, we need the following corollary of Lemma 5.35.

Corollary 5.36. *Let $A \subseteq \mathbb{F}_2^n$ be a set which satisfies $|A| \geq |\mathbb{F}_2^n| / r$. Then for every $\varepsilon > 0$ and every pair of functions $f, g : \mathbb{F}_2^n \to [0,1]$ there exists $t = O(\log(r/\varepsilon)/\varepsilon^2)$ and a a set X of size at least $|\mathbb{F}_2^n| / 2r^t$ such that for every set B s.t. $|B| \geq |\mathbb{F}_2^n| / r$ and every $(x_1, \ldots, x_\ell) \in X^\ell$,*

$$\mathop{\mathbb{E}}_{a\sim A, b\sim B}[f(a+b)] \approx_{\ell\varepsilon} \mathop{\mathbb{E}}_{a\sim A, b\sim B}[f(a+b+x_1+\cdots+x_\ell)]$$

and

$$\mathop{\mathbb{E}}_{a\sim A, b\sim B}[g(a+b)] \approx_{\ell\varepsilon} \mathop{\mathbb{E}}_{a\sim A, b\sim B}[g(a+b+x_1+\cdots+x_\ell)].$$

Proof. Assume by induction that

$$\mathop{\mathbb{E}}_{a\sim A, b\sim B}[f(a+b)] \approx_{(\ell-1)\varepsilon} \mathop{\mathbb{E}}_{a\sim A, b\sim B}[f(a+b+x_1+\cdots+x_{\ell-1})].$$

Since $|B + x_1 + \cdots + x_{\ell-1}| = |B| \geq |\mathbb{F}_2^n| / r$, by Lemma 5.35 we get

$$\mathop{\mathbb{E}}_{a\sim A, b\sim B}[f(a+b+x_1+\cdots+x_{\ell-1})] \approx_{\varepsilon} \mathop{\mathbb{E}}_{a\sim A, b\sim B}[f(a+b+x_1+\cdots+x_\ell)].$$

Then the claim follows by triangle inequality. The proof for the case of g is exactly the same. □

Now we are ready to prove Lemma 5.29.

Proof of Lemma 5.29. Define $\ell = \log(2r/\varepsilon)$. By Corollary 5.36 there exists $t = O(\ell^3/\varepsilon^2)$ and a set X of size $|\mathbb{F}_2^n| / 2r^t$ s.t. for every $(x_1, x_2, \ldots, x_\ell) \in X^\ell$,

$$\mathbb{E}[f(\mathbf{A}+\mathbf{B})] \approx_{\varepsilon/2} \mathbb{E}[f(\mathbf{A}+\mathbf{B}+x_1+\cdots+x_\ell)] \tag{5.5}$$

and

$$\mathbb{E}[g(\mathbf{A}+\mathbf{B})] \approx_{\varepsilon/2} \mathbb{E}[g(\mathbf{A}+\mathbf{B}+x_1+\cdots+x_\ell)].$$

Let $\mathbf{X}_1, \dots, \mathbf{X}_\ell$ be independent uniform distributions over X. Let $V = \mathrm{Spec}_{1/2}(X)^\perp$ and $\mathbf{V}$ be uniform distribution over V. Note that by Chang's lemma (Lemma 2.43), V has dimension at least $k' = m - O(\log(r)\log^3(r/\varepsilon)/\varepsilon^2) \geq k - O(\log(r)\log^3(r/\varepsilon)/\varepsilon^2)$. By Lemma 2.41, 2.40 and 2.36,

$$\mathbb{E}\left[f(\mathbf{A}+\mathbf{B}+\mathbf{X}_1+\cdots+\mathbf{X}_\ell)\right] = \sum_{\alpha\in\mathbb{F}_2^n} \widehat{\mu_A}(\alpha)\widehat{\mu_B}(\alpha)(\widehat{\mu_X}(\alpha))^\ell \widehat{f}(\alpha)$$

and

$$\mathbb{E}\left[f(\mathbf{A}+\mathbf{B}+\mathbf{X}_1+\cdots+\mathbf{X}_\ell+\mathbf{V})\right] = \sum_{\alpha\in\mathbb{F}_2^n} \widehat{\mu_A}(\alpha)\widehat{\mu_B}(\alpha)(\widehat{\mu_X}(\alpha))^\ell \widehat{\mu_V}(\alpha)\widehat{f}(\alpha).$$

Define $\mathbf{T} = \mathbf{A}+\mathbf{B}+\mathbf{X}_1+\cdots+\mathbf{X}_\ell$. Then

$$\begin{aligned}
\left|\mathbb{E}\left[f(\mathbf{T})\right] - \mathbb{E}\left[f(\mathbf{T}+\mathbf{V})\right]\right| &= \left|\sum_{\alpha\notin V^\perp} \widehat{\mu_A}(\alpha)\widehat{\mu_B}(\alpha)(\widehat{\mu_X}(\alpha))^\ell \widehat{f}(\alpha)\right| \\
&\qquad \text{(by Lemma 2.42)} \\
&\leq 2^{-\ell} \sum_{\alpha\notin V^\perp} \left|\widehat{\mu_A}(\alpha)\widehat{\mu_B}(\alpha)\widehat{f}(\alpha)\right| \\
&\qquad \text{(by definition of } \mathrm{Spec}_{1/2}(X)) \\
&\leq 2^{-\ell} \sum_{\alpha\notin V^\perp} |\widehat{\mu_A}(\alpha)\widehat{\mu_B}(\alpha)| \\
&\qquad \text{(since } \left|\widehat{f}(\alpha)\right| \leq 1) \\
&\leq 2^{-\ell} \cdot \sqrt{\left(\sum_{\alpha\in\mathbb{F}_2^n} \widehat{\mu_A}(\alpha)^2\right)\left(\sum_{\alpha\in\mathbb{F}_2^n} \widehat{\mu_B}(\alpha)^2\right)} \\
&\qquad \text{(by Cauchy-Schwarz)} \\
&\leq 2^{-\ell} \cdot r = \varepsilon/2. \\
&\qquad \text{(by Parseval's identity (Lemma 2.36))}
\end{aligned}$$

By triangle inequality and (5.5), we get $\mathbb{E}\left[f(\mathbf{A}+\mathbf{B})\right] \approx_\varepsilon \mathbb{E}\left[f(\mathbf{T}+\mathbf{V})\right]$. The exact same proof can be used to show that $\mathbb{E}\left[g(\mathbf{A}+\mathbf{B})\right] \approx_\varepsilon \mathbb{E}\left[g(\mathbf{T}+\mathbf{V})\right]$. □

5.6.3 Minimax Argument

First we review von-Neuman's minimax theorem.

Lemma 5.37 (minimax theorem [vN28])**.** *Let $\mathcal{X} \subseteq \mathbb{R}^n, \mathcal{Y} \subseteq \mathbb{R}^m$ be convex sets. Then for every bilinear function $g : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$,*

$$\min_{x \in \mathcal{X}} \max_{y \in \mathcal{Y}} g(x, y) = \max_{y \in \mathcal{Y}} \min_{x \in \mathcal{X}} g(x, y).$$

Then we prove Lemma 5.30.

Corollary 5.38 (Lemma 5.30, restated)**.** *Let Ω be a finite set, $\mathcal{X}$ be a convex set of distributions on Ω, and $\mathbf{Y}$ be a distribution on Ω. If for every function $f : \Omega \to [0, 1]$ there exists $\mathbf{X}_f \in \mathcal{X}$ such that $\mathbb{E}[f(\mathbf{X}_f)] - \mathbb{E}[f(\mathbf{Y})] \leq \varepsilon$, then there exists $\mathbf{X}^* \in \mathcal{X}$ such that $\mathbf{Y} \approx_\varepsilon \mathbf{X}^*$.*

Proof. Let $\mathcal{F}$ denote the set of all the functions from Ω to $[0, 1]$. Note that a distribution $\mathbf{X}$ can be represented by a vector in $\mathbb{R}^{|\Omega|}$, where the coordinate indexed by $s \in \Omega$ is $\Pr[\mathbf{X} = s]$. A function $f : \Omega \to [0, 1]$ can also be represented by a vector in $\mathbb{R}^{|\Omega|}$, where the coordinate indexed by $s \in \Omega$ is $f(s)$. Observe that $\mathcal{F}$ is convex. Define the function $g : \mathcal{X} \times \mathcal{F} \to \mathbb{R}$ to be

$$g(\mathbf{X}, f) := \mathbb{E}[f(\mathbf{X})] - \mathbb{E}[f(\mathbf{Y})] = \left(\sum_{s \in \Omega} \Pr[\mathbf{X} = s] \cdot f(s) \right) - \mathbb{E}[f(\mathbf{Y})].$$

Observe that g is bilinear. By minimax theorem,

$$\min_{\mathbf{X} \in \mathcal{X}} \max_{f \in \mathcal{F}} g(\mathbf{X}, f) = \max_{f \in \mathcal{F}} \min_{\mathbf{X} \in \mathcal{X}} g(\mathbf{X}, f) \leq \max_{f \in \mathcal{F}} (\mathbb{E}[f(\mathbf{X}_f)] - \mathbb{E}[f(\mathbf{Y})]) \leq \varepsilon.$$

That is, there exists $\mathbf{X}^* \in \mathcal{X}$ such that for every function $f : \Omega \to [0, 1]$, $\mathbb{E}[f(\mathbf{X}^*)] - \mathbb{E}[f(\mathbf{Y})] \leq \varepsilon$. If we take $f = \mathbb{1}_T$ for some $T \subseteq \Omega$, then $\mathbb{E}[f(\mathbf{X}^*)] - \mathbb{E}[f(\mathbf{Y})]$ is exactly $\Pr[\mathbf{X}^* \in T] - \Pr[\mathbf{Y} \in T]$. Therefore by definition of statistical distance, $\mathbf{X}^* \approx_\varepsilon \mathbf{Y}$. □

Chapter 6

Lower Bounds for Linear Branching Programs

In this chapter, we show an application of our sumset extractors on circuit lower bounds. Our main result in this chapter is that a sumset extractor requires almost maximum circuit size to compute in a model called read-once linear branching programs, a model that was recently introduced by Gryaznov, Pudlák and Talebanfard [GPT22].

6.1 Introduction

The central goal of complexity theory is to understand the power and limitation of different computation models. Motivated by this goal, it is natural to study the *lower*

This chapter is based on the following joint work:

- [CL23] Eshan Chattopadhyay and Jyun-Jie Liao. Hardness against linear branching programs and more. In *Leibniz International Proceedings in Informatics (LIPIcs): 38th Computational Complexity Conference (CCC 2023)*. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2023

bound problem: given a computation model and a corresponding complexity measure, can we find an explicit function (e.g. computable in polynomial time) that has large complexity? Researchers have studied this problem on many interesting circuit models such as bounded-depth circuits (AC^0), DeMorgan formula and branching programs, and many interesting results have been found. For example, one of the most notable results in this field is that it requires exponential number of gates to compute parity in AC^0 [Yao85, Has89]. (See the excellent book by Jukna [Juk12] for more about circuit lower bound problems.)

Interestingly, circuit lower bound problems have found connections with randomness extraction. For example, the state-of-the-art lower bound for Boolean circuits is based on affine extractors [LY22], which are extractors that work for weak sources that are uniform over affine subspaces. Affine extractors were also used to obtain almost optimal lower bounds for DNF of parities and parity decision trees [CS16b]. As another example, extractors for sources sampled by small-space algorithms [KRVZ11] were shown to be average-case hard against read-once branching program [CGZ22].

The main idea behind this connection is as follows. Suppose one can show that for every function $f : \mathcal{X} \to \{0,1\}$ with small complexity measure, the uniform distribution over the larger pre-image (say, $f^{-1}(0)$) is a source $\mathbf{X}$ with some specific structure. If one can construct an extractor for weak sources with this structure, then $f(\mathbf{X})$ is a constant while $\mathrm{Ext}(\mathbf{X})$ is close to uniform, immediately implying that f and Ext cannot be the same function. In fact, f cannot even approximately compute Ext much better than a random guess, i.e., Ext exhibits *average-case hardness* against f. For instance, to derive average-case lower bounds for parity decision trees, for which it is not hard to see that the pre-image is a disjoint union of affine subspaces, one can choose Ext to be an affine extractor. However, the choice of the extractor is

not always obvious. For example, the connection between general Boolean circuits and affine extractors [DK11, FGHK16, LY22] is more non-trivial.

In this chapter, we study the lower bound problem for *read-once linear branching programs* [GPT22]. Our main contribution is a new connection between lower bounds for read-once linear branching programs and sumset extractors.

6.1.1 Linear branching programs

Read-once linear branching programs (ROLBPs) were first studied by Gryaznov, Pudlák and Talebanfard [GPT22], motivated by its connection to proof complexity. Roughly speaking, a ROLBP is a read-once branching program that can make linear queries. We leave the definition of "read-once" for later and define a linear branching program first.

Definition 6.1 (Linear branching program [GPT22])**.** *A linear branching program on $\mathbb{F}_2^n$ is a directed acyclic graph P with the following properties:*

- *There is only one source s in P.*
- *There are two sinks in P, labeled with 0 and 1 respectively.*
- *Every non-sink node v is labeled with a linear function $\ell_v : \mathbb{F}_2^n \to \mathbb{F}_2$, which is called the* query *on node v. Moreover, there are exactly two outgoing edges from v, one is labeled with 1 and the other is labeled with 0.*

The size *of P is the number of non-sink nodes in P. We say P computes a boolean function $f : \mathbb{F}_2^n \to \{0,1\}$ in the following way. For every input $x \in \mathbb{F}_2^n$, we define the* computation path *of x as starting from s, and when on a node v which is not a sink, moving to the next*

node following the edge with label $\ell_v(x) \in \{0,1\}$. We repeat this process until the path ends at a sink. $f(x)$ is defined to be the label on this sink.[14]

The most natural definition of "read-once" for a linear branching program is that the queries made on every path is linearly independent. In this chapter, we focus on a more restricted model called *strongly read-once*.[15]

Definition 6.2 (Strongly Read-Once [GPT22])**.** *For every node v in a branching program P, define Pre_v to be the span of all queries that appear on any path from the source to v, and Post_v to be the span of all queries that appear on any path from v to a sink. (For every non-sink node v, both Pre_v and Post_v include ℓ_v. For any sink w we define $\mathsf{Post}_w = \{0\}$.) We say P is* strongly read-once *if the following two properties hold.*

- *For every edge $e = (u \to v)$, $\mathsf{Pre}_u \cap \mathsf{Post}_v = \{0\}$.*
- *For every non-sink node v, $\mathsf{Pre}_v \cap \mathsf{Post}_v = \{0, \ell_v\}$.*

As pointed out in [GPT22], although being more restricted than the natural definition of read-once, strongly read-once linear branching programs still generalize two important models: parity decision trees and read-once branching programs. A parity decision tree is a decision tree which can make linear queries. This model was first defined by Kushilevitz and Mansour [KM93] for its connection with Fourier analysis, and has recently received attention because of its connections to special cases of the log-rank conjecture in communication complexity [TWXZ13, HHL18] and quantum query complexity [GTW21]. A read-once branching program is another generalization of decision tree such that different paths can share nodes,

[14] In this chapter, we abuse notation and also use P to denote the function computed by P.

[15] Our definition here is slightly more general than the original one in [GPT22], but we don't view it as a substantial difference. We choose the definition here merely for simpler notation in the proofs.

and can be used to model streaming algorithms and randomized small-space algorithms. Similar to how decision trees are generalized to parity decision trees, it is natural to study ROBPs with linear queries.

The lower bound problem we are trying to answer is the following:

Question 6.3. *For a function $f : \mathbb{F}_2^n \to \mathbb{F}_2$, let $\mathsf{ROLBP}(f)$ denote the smallest possible size of a strongly read-once linear branching program that computes f. Can we find an explicit function f which is computable in polynomial time such that $\mathsf{ROLBP}(f)$ is as large as possible?*

Note that every function f has a trivial size upper bound $\mathsf{ROLBP}(f) \leq 2^n$ (e.g. a trivial decision tree of depth n), so our goal is to find a function f such that $\mathsf{ROLBP}(f)$ is as close to 2^n as possible. We are also interested in answering the average-case lower bound problem:

Question 6.4. *For a function $f : \mathbb{F}_2^n \to \mathbb{F}_2$ and any $\varepsilon > 0$, let $\mathsf{ROLBP}_\varepsilon(f)$ denote the smallest size of strongly read-once linear BP P such that*

$$\Pr_{x \sim \mathbb{F}_2^n}[P(x) = f(x)] \geq \frac{1}{2} + \varepsilon.$$

Can we find a function f which is computable in polynomial time such that $\mathsf{ROLBP}_\varepsilon(f)$ is as large as possible?

To obtain a lower bound for strongly read-once linear branching programs, [GPT22] introduced a new type of extractor called *directional affine extractors*.

Definition 6.5 (Directional Affine Extractor [GPT22])**.** *We say $\mathrm{DAExt} : \mathbb{F}_2^n \to \mathbb{F}_2$ is a (d, ε)-directional affine extractor if for any affine source $\mathbf{X} \in \mathbb{F}_2^n$ with min-entropy d, and any non-zero vector $a \in \mathbb{F}_2^n$, it holds that*

$$\mathrm{DAExt}(\mathbf{X} + a) + \mathrm{DAExt}(\mathbf{X}) \approx_\varepsilon \mathbf{U}_1.$$

[GPT22] proved that a directional affine extractor for small dimension has a large average-case lower bound for strongly read-once linear BP.

Theorem 6.6 ([GPT22, Theorem 17])**.** *Let* DAExt *be a* (d, ε)*-directional affine extractor. Then*

$$\mathsf{ROLBP}_{\sqrt{2\varepsilon}}(\mathrm{DAExt}) \geq \varepsilon 2^{n-d-1}.$$

In [GPT22], they constructed a directional affine extractor for dimension $(2/3 + o(1))n$, which implied a $2^{n/3-o(n)}$ average-case lower bound for ROLBPs. A natural open question left in [GPT22] was to construct a directional affine extractor for dimension $d = o(n)$, which would directly imply a $2^{n-o(n)}$ average-case lower bound for ROLBPs. However, this seems like a challenging problem. Indeed, even constructing affine extractors for dimension $d = o(n)$ has been a difficult task that has been recently resolved [Li16, CGL21]; a directional affine extractor is an affine extractor with additional *non-malleable* properties (see Theorem 6.11) and it is not clear how to use known techniques to construct such extractors for low dimension.

6.1.2 Our Results

In this chapter, we show that to get an average-case lower bound strongly read-once linear BP, it suffices to construct a *sumset extractor*. Our main theorem is as follows:

Theorem 6.7. *Let* SumExt *be a* (k_1, k_2, ε)*-sumset extractor. Then*

$$\mathsf{ROLBP}_{9\varepsilon}(\mathrm{SumExt}) > 2^{n-k_1-k_2-2}.$$

By taking SumExt to be the extractor in Theorem 5.2, we improve the best lower bound for strongly read-once linear BP from $2^{n/3-o(n)}$ to $2^{n-o(n)}$, which is almost optimal.

Theorem 6.8. *There is a function* SumExt *which can be computed in polynoimal time such that*

$$\mathsf{ROLBP}_{n^{-\Omega(1)}}(\mathrm{SumExt}) > 2^{n-\mathrm{polylog}(n)}.$$

In addition, we also show some relations between directional affine extractors and other extractors, which might be of independent interest. First we show that a directional affine extractor can extract from a special case of sumset sources. In addition, we note that the proof is just a direct application of Leftover Hash Lemma [ILL89].

Theorem 6.9. *Let* DAExt *be a* (d, ε)*-directional affine extractor. Then for any* $\mathbf{B} \in \mathbb{F}_2^n$ *which is uniform over an affine subspace of dimension* d*, and any* $\mathbf{A} \in \mathbb{F}_2^n$ *independent of* $\mathbf{B}$ *such that* $\mathrm{H}_\infty(\mathbf{A}) \geq \log(1/\varepsilon) + 1$*,*

$$(\mathrm{DAExt}(\mathbf{A} + \mathbf{B}) \approx_{\sqrt{\varepsilon/2}} \mathbf{U}_1) \mid \mathbf{B}.$$

This theorem together with a structural lemma we use to prove Theorem 6.7 implies a modular way to prove Theorem 6.6.

Secondly, we show that a directional affine extractors is equivalent to a special case of seedless non-malleable extractors [CG14].

Definition 6.10. *We say* $\mathrm{Ext} : \mathbb{F}_2^n \to \{0,1\}$ *is a* (d, ε)-non-malleable affine extractor against shifts *if for every source* $\mathbf{X} \in \mathbb{F}_2^n$ *which is uniform over an affine subspace of dimension* d*, and every non-zero shift* $a \in \mathbb{F}_2^n$*,*

$$(\mathrm{Ext}(\mathbf{X}) \approx_\varepsilon \mathbf{U}_1) \mid \mathrm{Ext}(\mathbf{X} + a).$$

It's easy to see that a (d, ε)-non-malleable affine extractor against shifts is also a (d, ε)-directional affine extractor. In this chapter we prove that the converse is also true.

Theorem 6.11. *For every $d \in \mathbb{N}, \varepsilon > 0$ such that $d \geq \log(1/\varepsilon)$, a (d, ε)-directional affine extractor is also a $(d, O(\sqrt{\varepsilon}))$-non-malleable affine extractor against shifts.*

6.1.3 Pseudorandomness against Linear Branching Programs

Motivated by close connections between hardness and pseudorandomness [Nis94], we initiate the study of obtaining pseudorandomness results against linear branching programs. Generally speaking, in the pseudorandomness problem for a function class $\mathcal{F}$, our goal is to construct a *pseudorandom distribution* which can be generated with only $r \ll n$ random bits but is indistinguishable from the n-bit uniform distribution $\mathbf{U}_n$ by any function in $\mathcal{F}$. We now formally define a hitting set generator (HSG), which is the one-sided variant of a pseudorandom generator (PRG).

Definition 6.12 (Hitting Set Generators)**.** *We say a set $H \subseteq \{0,1\}^n$ is a* hitting *set with error ε for a class of functions $\mathcal{F}$ (on n-bit input), if for every function $f : \{0,1\}^n \to \{0,1\}$ in $\mathcal{F}$ such that $\Pr_{x \sim \mathbf{U}_n}[f(x) = 1] \geq \varepsilon$, there exists $h \in H$ such that $f(h) = 1$. Moreover, $G : \{0,1\}^d \to \{0,1\}^n$ is called a* hitting set generator *(HSG) with error ε for a class of functions $\mathcal{F}$ if $\{G(s)\}_{s \in \{0,1\}^d}$ is a hitting set for $\mathcal{F}$, and s is called the seed length of G.*

Constructing good HSGs for (standard) read-once branching programs is a central problem in complexity theory. If one can construct an explicit HSG with seed length $O(\log(n))$ and $O(1)$ error for ROBPs of size $\mathrm{poly}(n)$, this would imply $\mathbf{RL} = \mathbf{L}$, which is a major open problem in complexity theory. Interestingly, it was recently shown [CH20] that HSGs suffice to even derandomize $\mathbf{BPL}$.

We note that for the above derandomization applications, it suffices to construct a HSG for oblivious ROBPs with ordered input. That is, given an n-bit input $x = (x_1, \ldots, x_n)$, the ROBPs read the bits $x_1, x_2, \ldots, x_n$ in order, regardless of what x is.

For oblivious ROBPs with ordered input, the best known construction is due to Nisan [Nis92] that has seed length $O(\log^2(n))$ (which in fact is a pseudorandom generator). However, in spite of the improvement in several restricted sub-classes of ROBPs, Nisan's result remains the best known construction in the general setting after three decades of work.

A recent research direction has been to find approaches that are completely different from Nisan's construction. In this direction, researchers considered the task of constructing PRGs (and HSGs) for a natural generalization of ROBPs called as (oblivious) unordered ROBPs for which it is known that Nisan's construction fails to work [Tzu09, BPW11]. An unordered ROBPs still read the bits of x in a fixed order that does not depend on x, but this order is unknown. By an impressive line of work culminating with a beautiful construction by Forbes and Kelley [FK18], we now have explicit PRGs with seed length $O(\log^3(n))$ for unordered ROBPs. The general approach used to construct PRGs for this model is based on analyzing the effects of random restrictions on ROBPs and leveraging bounds on the Fourier spectrum of branching programs [RSV13, CHRT18].

This gives us further motivation to study pseudorandomness against *oblivious ROLBPs*, which is a further generalization of unordered ROBPs.

Definition 6.13 (Oblivious ROLBPs)**.** *We say a read-once linear branching programs P on input $\mathbb{F}_2^n$ is oblivious if the nodes can be divided into layers $L_0, \ldots, L_n$ such that*

- *L_0 only contains the source, and L_n consists of all the sinks.*
- *For every $0 \leq i < n$, every edge from nodes in L_i connects to a node in L_{i+1}.*
- *For every $0 \leq i < n$, every node on L_i is labeled with the same linear query ℓ_i.*
- *$(\ell_0, \ldots, \ell_{n-1})$ is a basis of $\mathbb{F}_2^n$.*

The width *of P is defined as* $\max_{i\in[n]}(|L_i|)$.

We note that unordered ROBPs correspond to the case of $(\ell_0, \ldots, \ell_{n-1})$ being a permutation of the standard basis. Thus, Nisan's PRG construction fails to work for oblivious ROLBPs. Further, it is not clear how to use the techniques of random restriction based constructions employed for unordered ROBPs when the layers can be arbitrary linear functions. Thus, it looks like we need new ideas to obtain pseudorandomness against oblivious ROLBPs.

Our first observation is that the case of width $w = 2$ is easy since it is well known that a small-biased distribution [NN93, AGHP92, TS17] fools such programs.[16] This follows since small-biased distributions are invariant under full-rank linear transformations. Further, [BDVY13] proved that sum of small-biased distributions fools width-2 ROBPs that reads multiple bits. Thus, one can obtain a similar result for the linear analogue of these programs. It has been asked by Vadhan and Reingold (see [LV17]) whether sums of small-biased distributions can be employed to construct PRGs (or HSG) for general ROBPs. Indeed a positive answer to this question would immediately imply a PRGs (or HSG) for oblivious ROLBPs. We are not able to resolve this conjectured approach, and take a different route that we describe below.

We take an initial step towards constructing HSGs against oblivious ROLBPs of width more than 2, and focus on the sub-class of regular oblivious ROLBPs. A regular linear branching program is a linear branching program in which every non-source node has in-degree 2. As our main result here, we construct a *hitting set generator* with $(1 - \Omega(1))n$ seed length for *regular* oblivious ROLBPs with *constant width*.

Theorem 6.14. *For every $w \in \mathbb{N}$, there is an explicit construction of HSG for regular*

[16] This result is due to Saks and Zuckerman. See [BDVY13] for sketch of a proof.

oblivious ROLBPs of width w with seed length $(w-1)+\lceil(1-2^{-(w-1)})n\rceil$.

Interestingly, our construction is based on a well-studied problem called rank-r Kakeya set [EOT10, KLSS11], which is a set that contains a r-dimensional affine subspace in every direction.

Definition 6.15. *A set $K \subseteq \mathbb{F}_2^n$ is called a* rank-r Kakeya set *(over $\mathbb{F}_2^n$) if for every r-dimension subspace $V \subseteq \mathbb{F}_2^n$, there exists $b \in \mathbb{F}_2^n$ such that $V + b \subseteq K$.*

We prove the following theorem.

Theorem 6.16. *A rank-r Kakeya set is a hitting set for oblivious read-once regular linear BP of width $(r+1)$.*

To get an efficiently computable HSG, we take the following simple construction of rank-r Kakeya set constructed by Kopparty, Lev, Saraf and Sudan [KLSS11].

Theorem 6.17 ([KLSS11])**.** *For every $r, n \in \mathbb{N}$ s.t. $r \leq n$, there is an explicit construction of rank-r Kakeya set $K_{n,r} \subseteq \mathbb{F}_2^n$ with size at most $2^{\lceil(1-2^{-r})n\rceil+r}$, which is defined as follows. Let $I_1, \ldots, I_{2^r}$ be a parition of $[n]$, each having size at least $\lfloor 2^{-r}n \rfloor$. Then*

$$K_{n,r} = \bigcup_{t=1}^{2^r} \mathrm{span}\left(\{e_i\}_{i \notin I_i}\right).^{17}$$

In other words, $K_{n,r}$ is the union of 2^r boolean subcubes where the i-th subcube contains every point $x \in \mathbb{F}_2^n$ such that the x_{I_i} is 0.

To prove Theorem 6.14, observe that we can construct an efficient HSG with seed length $r + \lceil(1-2^{-r})n\rceil$ that uses the first r bits to select a set I_i and use the remaining $\lceil(1-2^{-r})n\rceil \geq n - |I_i|$ bits to choose a point in the subcube corresponding to I_i.

[17] Here e_i denote a the i-th standard basis vector in $\mathbb{F}_2^n$ which has its i-th coordinate being 1 and other coordinates being 0.

We note our approach based on Kakeya set does not seem to extend beyond regular ROLBPs. For non-regular oblivious ROLBPs, we observe that the construction in Theorem 6.17 is not a hitting set for width 3, because a read-once CNF $\bigwedge_{t=1}^{2^r}(\bigvee_{i\in I_t} x_i)$ always outputs 0 on $K_{n,r}$, and a read-once CNF can be computed by a width-3 ROBP.

Further, while one might hope to extend our result to larger width (for regular ROLBPs) with a better construction of Kakeya sets, we show that the construction in Theorem 6.17 is essentially optimal. This negative result also answers an open question in [KLSS11] (for the case of $\mathbb{F}_2^n$), where they asked whether there is a better construction of rank-r Kakeya sets than Theorem 6.17. This lower bounds may be of independent interest.

Theorem 6.18. *Every rank-r Kakeya set over $\mathbb{F}_2^n$ has size at least $2^{(1-2^{-r})(n+2)-r}$.*

Subsequent Works. As mentioned in Chapter 5, Li [Li23] recently improved the entropy requirement of explicit sumset extractors to $O(\log(n))$ in the constant error regime. By Theorem 6.7, such an extractor implies a $2^n/\operatorname{poly}(n)$ average-case lower bound with constant correlation. Another work by Li and Zhong [LZ23] showed how to construct a directional affine extractor DAExt with $2^{-n^{\Omega(1)}}$ for $o(n)$ entropy. As proved in [GPT22], this implies an average-case lower bound of size $2^{n-o(n)}$ and exponentially small correlation, i.e., $\mathsf{ROLBP}_{2^{-n^{\Omega(1)}}}(\mathrm{DAExt}) \geq 2^{n-o(n)}$.

Organization. In Section 6.2 we prove our lower bound for ROLBPs (Theorem 6.7). In Section 6.3 we prove Theorem 6.9 and Theorem 6.11. In Section 6.4 we discuss our results about hitting sets for ROLBPs.

6.2 ROLBP Lower Bounds based on Sumset Extractors

In this section, we prove Theorem 6.7. We first discuss the main ideas behind the proof before formally proving it. Given a read-once linear BP $P : \mathbb{F}_2^n \to \{0,1\}$ and any $b \in \{0,1\}$, the uniform distribution over the pre-image $P^{-1}(b)$ corresponds to the uniform distribution over all the computation path from the source s to the sink labeled b. For every edge e, whether a computation path pass goes through e and ends at a sink labeled b can be divided into two events: whether a path starting from s would reach e, and whether a path starting from e would end at a sink labeled b. The strongly read-once property guarantees that we can divide $\mathbb{F}_2^n$ into two complemented subspaces V_A, V_B such that the first event is determined by the projection of the input $x \in \mathbb{F}_2^n$ on V_A, and the second event is determined by the projection of x on V_B. Given a uniform input $\mathbf{X} \in \mathbb{F}_2^n$, the two projections are independent. Therefore, conditioned on the computation path passing through e and end at a sink labeled b, $\mathbf{X}$ can be written as the sum of two independent sources $\mathbf{A}+\mathbf{B}$, where $\mathrm{Supp}(\mathbf{A}) \subseteq V_A$ and $\mathrm{Supp}(\mathbf{B}) \subseteq V_B$. It remains to choose a cut E such that for every choice of $e \in E$, the two sources $\mathbf{A}, \mathbf{B}$ stated above both have enough entropy.

We formalize the ideas above as the following structural lemma:

Lemma 6.19. *Let $\mathbf{X}$ be a uniform random variable over $\mathbb{F}_2^n$. For every strongly read-once linear BP $f : \mathbb{F}_2^n \to \{0,1\}$ of size s and every $d \in [n]$, there exists a random variable $\mathbf{E}$, and random variables $\mathbf{A}, \mathbf{B} \in \mathbb{F}_2^n$, s.t.*

- $\mathbf{E}$ *has support size at most* $2s$.
- $\mathbf{X} = \mathbf{A} + \mathbf{B}$.
- $\mathbf{A} \leftrightarrow \mathbf{E} \leftrightarrow \mathbf{B}$ *forms a Markov chain*
- $\mathbf{B}\,|_{\mathbf{E}=e}$ *is an affine source with min-entropy d for every $e \in \mathrm{Supp}(\mathbf{E})$.*

- *There is a deterministic function g s.t. $g(\mathbf{E}, \mathbf{B}) = f(\mathbf{X})$.*

Proof. We show that there exist some functions E, A, B s.t. $\mathbf{E} = E(\mathbf{X}), \mathbf{A} = A(\mathbf{X}), \mathbf{B} = B(\mathbf{X})$ satisfy the above claim. Fix any $x \in \mathbb{F}_2^n$. Consider the computation path of x, and let v be the first node on this path which satisfies that $\dim(\mathsf{Post}_v) \leq d$. Note that v is well-defined because the last node w on this path satisfies $\dim(\mathsf{Post}_w) = 0 \leq d$. Then we define $E(x) := (u \to v)$ to be the edge right before v in this path. (If v is the source, we define u to be a dummy node $\perp$, and define $\mathsf{Pre}_\perp = \{0\}$.) First we claim that $\dim(\mathsf{Pre}_u) \leq n - d$. If $u = \perp$ then the claim is trivially true. Otherwise, observe that $\dim(\mathsf{Post}_u) \geq d + 1$ by the definition of v, and by the strongly read-once property we have

$$\dim(\mathsf{Pre}_u) \leq n + \dim(\mathsf{Pre}_u \cap \mathsf{Post}_u) - \dim(\mathsf{Post}_u) \leq n + 1 - (d+1) = n - d.$$

Observe that $\mathsf{Pre}_u \cap \mathsf{Post}_v = \{0\}$ by the strongly read-once property. Now we choose an arbitrary basis $(b_1, \ldots, b_n)$ of $\mathbb{F}_2^n$ such that $\mathrm{span}(\{b_i\}_{1 \leq i \leq \dim(\mathsf{Pre}_u)}) = \mathsf{Pre}_u$ and $\mathrm{span}(\{b_{n-i}\}_{0 \leq i < \dim(\mathsf{Post}_v)}) = \mathsf{Post}_v$. Define $\mathsf{Pre}'_u = \mathrm{span}(\{b_i\}_{1 \leq i \leq n-d})$ and $\mathsf{Post}'_v = \mathrm{span}(\{b_i\}_{n-d < i \leq n})$. Note that $\mathsf{Pre}_u \subseteq \mathsf{Pre}'_u$, $\mathsf{Post}_v \subseteq \mathsf{Post}'_v$ and Pre'_u and Post'_v are complemented subspaces. Then define $(A(x), B(x))$ to be the unique pair in $(\mathsf{Post}'_v)^\perp \times (\mathsf{Pre}'_u)^\perp$ s.t. $A(x) + B(x) = x$. It remains to prove that $\mathbf{E} = E(\mathbf{X}), \mathbf{A} = A(\mathbf{X}), \mathbf{B} = B(\mathbf{X})$ satisfy our claim.

First it's easy to see that the support size of $\mathbf{E}$ is upper bounded by $2s$: if the source s satisfies $\dim(\mathsf{Post}_s) \leq d$, v is always the source s and $\mathbf{E}$ has support size 1; otherwise $\mathbf{E}$ is an edge in the branching program, and there are at most $2s$ choices. Moreover, $\mathbf{X} = \mathbf{A} + \mathbf{B}$ by definition of A and B. To prove the remaining claims, consider any possible fixing $\mathbf{E} = e := (u \to v)$. Let $(A_e(x), B_e(x))$ denote the unique pair in $(\mathsf{Post}'_v)^\perp \times (\mathsf{Pre}'_u)^\perp$ s.t. $A_e(x) + B_e(x) = x$. We claim that there exists a set

$S \subseteq (\mathsf{Post}'_v)^\perp$ so that $E(x) = e$ if and only if $A_e(x) \in S$. This implies that $(\mathbf{A}, \mathbf{B})|_{\mathbf{E}=e}$ is exactly the uniform distributions over $S \times (\mathsf{Pre}'_u)^\perp$, which satisfies the third and fourth claim. To prove this claim, observe that whether $E(x) = e$ can be decided by the following procedure. We follow the computation path of x, but stop and answer "NO" if we reach any node w such that either w cannot reach u (so that $E(x)$ can never be e regardless of the remaining queries) or $\dim(\mathsf{Post}_w) \leq d$ (so that $E(x)$ would be the edge ending at w instead of e). Otherwise, if we reach the edge e we stop and answer "YES". Observe that every linear query ℓ we made in this procedure is in Pre_u. Moreover, for every such query, $\ell(x) = \ell(A_e(x)) + \ell(B_e(x)) = \ell(A_e(x))$ because $B_e(x) \in (\mathsf{Pre}'_u)^\perp \subseteq (\mathsf{Pre}_u)^\perp$. Therefore, the event $E(x) = e$ is completely determined by $A_e(x)$, which proves our claim. Finally, observe that conditioned on $E(x) = e$, the value of $f(x)$ is determined by queries in Post_v, and every such query ℓ satisfies that $\ell(x) = \ell(A_e(x)) + \ell(B_e(x)) = \ell(B_e(x)) = \ell(B(x))$ because $A_e(x) \in (\mathsf{Post}'_v)^\perp \subseteq (\mathsf{Post}_v)^\perp$. Therefore by choosing $g(e, \cdot)$ to be the subprogram of f starting at v, the last condition is also satisfied. □

Now we are ready to prove Theorem 6.7.

Proof of Theorem 6.7. Let SumExt be an extractor for (n, k_1, k_2)-sumset source with error ε, and let $f : \mathbb{F}_2^n \to \{0, 1\}$ be any strongly read-once linear BP of size $s = 2^{n-k_1-k_2-2}$. Let $\mathbf{X}$ be a uniform random variable over $\mathbb{F}_2^n$. We want to show that

$$(\mathrm{SumExt}(\mathbf{X}), f(\mathbf{X})) \approx_{9\varepsilon} (\mathbf{U}_1, f(\mathbf{X})), \tag{6.1}$$

which would imply $\Pr_{x \sim \mathbf{X}}[f(x) = \mathrm{SumExt}(x)] \leq \frac{1}{2} + 9\varepsilon$ for every f of size s, and hence $\mathsf{ROLBP}_{9\varepsilon}(\mathrm{SumExt}) > s$.

Let $\mathbf{E}, \mathbf{A}, \mathbf{B}$ be the random variables depending on $\mathbf{X}$ as in Lemma 6.19, by taking $d = k_2 + 1$. Recall that $\mathbf{E}, \mathbf{A}, \mathbf{B}$ have the following properties:

- $\mathbf{E}$ has support size at most $2s$.
- $\mathbf{X} = \mathbf{A} + \mathbf{B}$.
- $\mathbf{A} \leftrightarrow \mathbf{E} \leftrightarrow \mathbf{B}$ forms a Markov chain
- $\mathbf{B}\,|_{\mathbf{E}=e}$ is an affine source with min-entropy d for every $e \in \mathrm{Supp}(\mathbf{E})$.
- There is a deterministic function g s.t. $g(\mathbf{E}, \mathbf{B}) = f(\mathbf{X})$.

Therefore we can rewrite Equation (6.1) as

$$(\mathrm{SumExt}(\mathbf{A} + \mathbf{B}) \approx_{9\varepsilon} \mathbf{U}_1) \mid g(\mathbf{E}, \mathbf{B}). \tag{6.2}$$

Consider the function $\mathrm{Ext} : (\mathbb{F}_2^n)^2 \to \{0,1\}$ defined as $\mathrm{Ext}(a, b) = \mathrm{SumExt}(a + b)$. We claim that for every $e \in \mathrm{Supp}(\mathbf{E})$, we have that

$$(\mathrm{Ext}(\mathbf{A}', \mathbf{B}|_{\mathbf{E}=e}) \approx_{3\varepsilon} \mathbf{U}_1) \mid g(e, \mathbf{B}|_{\mathbf{E}=e})$$

for any source $\mathbf{A}'$ with min-entropy k_1. This would imply (6.2) because of the following. Observe that

$$\widetilde{\mathrm{H}}_\infty(\mathbf{A} \mid \mathbf{E}) = \widetilde{\mathrm{H}}_\infty(\mathbf{A} \mid (\mathbf{B}, \mathbf{E})) \geq \widetilde{\mathrm{H}}_\infty((\mathbf{A}, \mathbf{B}) \mid \mathbf{E}) - d \geq (n - \log(2s)) - d \geq k_1,$$

where the first equality is by the fact that $\mathbf{A}$ and $\mathbf{B}$ are independent conditioned on $\mathbf{E}$, and the first and second inequalities are by chain rule (Lemma 2.10). Furthermore, we can w.l.o.g. assume that $k_1 \leq n - 1$ (since otherwise the bound is trivial). Therefore we can apply Lemma 2.23 on Ext to get Equation (6.2).

Next we prove the claim. Let $\mathbf{A}' \in \mathbb{F}_2^n$ be any distribution such that $\mathrm{H}_\infty(\mathbf{A}') \geq k_1$. By definition of SumExt, we have that for every random variable $\mathbf{B}'$ such that $\mathrm{H}_\infty(\mathbf{B}') \geq k_2$,

$$\mathrm{SumExt}(\mathbf{A}' + \mathbf{B}') \approx_\varepsilon \mathbf{U}_1.$$

In addition, by chain rule, $\widetilde{\mathrm{H}}_\infty(\mathbf{B}_e \mid g(e, \mathbf{B}_e)) \geq \mathrm{H}_\infty(\mathbf{B}_e) - 1 = k_2$. Therefore, again by Lemma 2.23 we can conclude that

$$(\mathrm{SumExt}(\mathbf{A}' + \mathbf{B}_e) \approx_{3\varepsilon} \mathbf{U}_1) \mid g(e, \mathbf{B}_e),$$

and this is exactly what we claimed. □

6.3 Directional Affine Extractors

In this section we prove the connection between directional affine extractors and other extractors (Theorem 6.9 and Theorem 6.11).

Proof of Theorem 6.9. Observe that for every distinct $a_1, a_2 \in \mathbb{F}_2^n$,

$$\begin{aligned}\Pr_{b \sim \mathbf{B}}[\mathrm{DAExt}(a_1 + b) = \mathrm{DAExt}(a_2 + b)] &= \Pr_{b \sim \mathbf{B}}[(\mathrm{DAExt}(a_1 + b) + \mathrm{DAExt}(a_2 + b)) = 0] \\ &\leq \frac{1 + \varepsilon}{2},\end{aligned}$$

by definition of $(d, \varepsilon/2)$-directional affine extractor. This means the function $h(a, b) = \mathrm{DAExt}(a + b)$ is ε-almost universal over randomness $\mathbf{B}$. By leftover hash lemma (Lemma 2.22), h is a $(\log(1/\varepsilon) + 1, \sqrt{\varepsilon/2})$-strong extractor with seed $\mathbf{B}$. In other words, for every distribution $\mathbf{A} \in \mathbb{F}_2^n$ independent of $\mathbf{B}$ such that $\mathrm{H}_\infty(\mathbf{A}) \geq \log(1/\varepsilon) + 1$,

$$(\mathrm{DAExt}(\mathbf{A} + \mathbf{B}) \approx_{\sqrt{\varepsilon/2}} \mathbf{U}_1) \mid \mathbf{B}.$$

□

Note that this theorem gives an alternative proof of Theorem 6.6.

Alternative proof of Theorem 6.6. First we apply the structural lemma (Lemma 6.19) to get random variables $\mathbf{A}, \mathbf{B}, \mathbf{E}$ such that $\mathbf{X} = \mathbf{A} + \mathbf{B}$, $\mathbf{A} \leftrightarrow \mathbf{E} \leftrightarrow \mathbf{B}$, $f(\mathbf{X}) =$

$g(\mathbf{B}, \mathbf{E})$, and $\mathbf{B}|_{\mathbf{E}=e}$ is an affine source with min-entropy k for every $e \in \mathrm{Supp}(\mathbf{E})$. By Theorem 6.9, we know that for every distribution $\mathbf{A}'$ of min-entropy $1 + \log(1/\varepsilon)$,

$$(\mathrm{DAExt}(\mathbf{A}' + \mathbf{B}|_{\mathbf{E}=e}) \approx_{\sqrt{\varepsilon/2}} \mathbf{U}_1) \mid \mathbf{B}|_{\mathbf{E}=e},$$

which also implies

$$(\mathrm{DAExt}(\mathbf{A}' + \mathbf{B}|_{\mathbf{E}=e}) \approx_{\sqrt{\varepsilon/2}} \mathbf{U}_1) \mid g(\mathbf{B}|_{\mathbf{E}=e}, e).$$

In addition, if $s \leq \varepsilon 2^{n-d-3}$ then $\widetilde{\mathrm{H}}_\infty(\mathbf{A} \mid \mathbf{E}) \geq n - d - \log(4s) \geq 1 + \log(1/\varepsilon)$, which by Lemma 2.23 implies

$$(\mathrm{DAExt}(\mathbf{A}' + \mathbf{B}|_{\mathbf{E}=e}) \approx_{3\sqrt{\varepsilon/2}} \mathbf{U}_1) \mid g(\mathbf{B}|_{\mathbf{E}=e}, e).$$

Therefore $\mathsf{ROLBP}_{3\sqrt{\varepsilon/2}}(\mathrm{DAExt}) \geq \varepsilon 2^{n-d-3}$. □

Remark 6.20. *One can get the exact same constant factors as Theorem 6.6 by using the fact that Leftover Hash Lemma has no loss at all when we consider sources with average conditional min-entropy rather than sources with worst-case min-entropy.*

Next we prove Theorem 6.11.

Proof of Theorem 6.11. To prove this theorem, we need an extension of Vazirani's XOR lemma, which can be found in [DLWZ11, Lemma 3.8]. We only state the special case we need here.

Lemma 6.21. *Let $(\mathbf{W}, \mathbf{W}')$ be a random variable over $(\mathbb{F}_2)^2$. If $\mathbf{W} \approx_\varepsilon \mathbf{U}_1$ and $(\mathbf{W} + \mathbf{W}') \approx_\varepsilon \mathbf{U}_1$, then*

$$(\mathbf{W} \approx_{4\varepsilon} \mathbf{U}_1) \mid \mathbf{W}'.$$

With this lemma, it suffices to prove that for every (d, ε)-directional affine extractor $\mathrm{DAExt} : \mathbb{F}_2^n \to \{0,1\}$ the following holds. for every source $\mathbf{X} \in \mathbb{F}_2^n$ which is uniform over an affine subspace of dimension d, and every non-zero shift $a \in \mathbb{F}_2^n$,

- $\mathrm{DAExt}(\mathbf{X}) \approx_{\sqrt{\varepsilon}} \mathbf{U}_1$, and
- $\mathrm{DAExt}(\mathbf{X}) + \mathrm{DAExt}(\mathbf{X} + a) \approx_{\sqrt{\varepsilon}} \mathbf{U}_1$.

The second condition is directly implied by the definition of DAExt. It remains to prove the first condition. Let V be the linear subspace which is a shift of the affine subspace $\mathrm{Supp}(\mathbf{X})$, and let $\mathbf{V}$ denote the uniform distribution over V which is independent of $\mathbf{X}$. Observe that $\mathbf{V} + \mathbf{X}$ is the same distribution as $\mathbf{X}$, and $\mathrm{H}_\infty(\mathbf{V}) \geq d \geq \log(1/\varepsilon)$. Then, by Theorem 6.9 we have $\mathrm{DAExt}(\mathbf{X} + \mathbf{V}) \approx_{O(\sqrt{\varepsilon})} \mathbf{U}_1$. □

We note that Chattopadhyay and Li [CL17] considered the problem of constructing non-malleable extractors against the more general class of all linear functions, but their results requires to the affine source to have dimension $0.99n$. It seems difficult to extend their techniques to handle smaller min-entropy, even against the weaker class of shifts.

6.4 Kakeya Sets and HSGs for Regular ROLBPs

In this section, we prove Theorem 6.16, which says that rank-r Kakeya set is a hitting set for oblivious ROLBPs of width $(r+1)$, and Theorem 6.18, a size lower bound for rank-r Kakeya set over $\mathbb{F}_2^n$.

In [BRRY14], it was proved that a Hamming ball of radius $(w-1)$ is a hitting set for regular read-once branching program of width w.[18] Their proof relies on the fact that there are only $(w-1)$ "crucial layers" such that we can only make a "fatal

[18] A Hamming ball of radius r centered around $c \in \{0,1\}^n$ is the set of all the strings which are different from c in at most r bits.

decision" which goes from a "possibly accept" node to an "always reject" node in these layers. The formal statement is as follows.

Lemma 6.22 ([BRRY14]). *For a ROBP f on $\mathbb{F}_2^n$ with layers $L_0, L_1, \ldots, L_n$, we say a layer L_i is crucial if there exists $v \in L_i$ and an edge $(u \to v)$ such that u can reach an accepting state but v cannot.*[19] *Then for every $w \in \mathbb{N}$, a regular ROBP of width w has at most $(w-1)$ crucial layers.*

Based on this lemma, [BRRY14] observed that in order to find an input x of which the computation path reaches an accepting state, we only need to make sure that we do not make any fatal decision in the crucial layers, and the bits read in the other layer can simply be set to 0. Therefore, the Hamming ball of radius $(w-1)$ centered around 0 is a hitting set for regular ROBPs of width w, because the Hamming ball covers every possible decision in the crucial layers, no matter where the crucial layers are. This makes sure that we can find a string which does not make any fatal decision, and this string would reach the sink labeled with 1 in the end.

To generalize this argument to the setting of regular ROLBPs, we want to find a set H such that for every possible rotation R of $\mathbb{F}_2^n$, the rotation of H (denoted by $R(H)$) contains a string which does not make any fatal decision. A naive idea is to find a set which contains every possible rotation of Hamming balls centered at 0. However, this contains exactly the whole set $\mathbb{F}_2^n$. To deal with this issue, we observe that for the argument in [BRRY14] to work, we only need to make sure that for every possible choices of crucial layers $L_{i_1}, \ldots, L_{i_{w-1}}$, where $I = \{i_1, \ldots, i_{w-1}\} \subseteq [n]$, there exists a fixing of the bits outside the crucial layer, such that we enumerate over every possible choice of bits in the crucial layers. Note that the fixing does not need to be 0 and can depend on the choice of crucial layers I. That is, for every set $I \subseteq [n]$ of

[19] Accepting states are the sinks with label 1.

size at most $(w-1)$, we need to enumerate over a subcube with free bits in I and arbitrary fixing outside I. To ensure this for every possible rotation, what we need is exactly a *Kakeya set*. Next we give a formal proof of our argument.

Lemma 6.23. *Let $H \subseteq \mathbb{F}_2^n$ be a set which satisfies the following: for every $I \subseteq [n]$ of size $(w-1)$, there exists $b \in \mathbb{F}_2^n$ such that $b + \mathrm{span}(\{e_i\}_{i \in I}) \subseteq H$. Then H is a hitting set for regular branching programs of width w.*

Proof. Let f be a regular branching program of width w which accepts at least one string. By Lemma 6.22, there are at most $(w-1)$ crucial layers in f. Let I denote the set of indices of these crucial layers. By assumption there exists $b \in \mathbb{F}_2^n$ such that $b + \mathrm{span}(\{e_i\}_{i \in I}) \subseteq H$. Now we define a string $b' \in \mathbb{F}_2^n$ inductively as follows. Let v_0 be the source of f, and for every non-sink node v and every $b \in \{0,1\}$ let $\mathsf{next}(v,b)$ denote the node which v connects to with an edge of label b. For i from 1 to n, we define b'_i (the i-th bit of b') as follows:

- If $i \notin I$, then set $b'_i = b_i$.
- If $i \in I$, then set $b'_i = 0$ if $\mathsf{next}(v_{i-1}, 0)$ can reach a accepting state. Otherwise set $b'_i = 1$.

Then we define $v_i = \mathsf{next}(v_{i-1}, b'_i)$. First observe that b' only differ from b on the bits with indices in I. Therefore $b' \in H$. It remains to prove that $f(b') = 1$. Next we prove by induction that every v_i can reach a accepting state. This means v_n is a accepting state, i.e., $f(b') = 1$. For the base case, note that v_0 is the source and hence can reach a accepting state by assumption. To prove that v_i can reach an accepting state assuming that v_{i-1} can reach an accepting state, consider two cases. If $i \notin I$, then the i-th layer is not crucial, which means v_i can reach a accepting state. If $i \in I$, observe that at least one node in $\{\mathsf{next}(v_{i-1}, 0), \mathsf{next}(v_{i-1}, 1)\}$ should be able to reach

a accepting state, because they are the only nodes that v_{i-1} can connect to, and v_{i-1} can reach a accepting state. Therefore v_i can also reach a accepting state by definition of b'_i. □

Now we are ready to prove Theorem 6.16.

Proof of Theorem 6.16. Let K be a rank-r Kakeya set, and f be any oblivious ROLBP of width $(r+1)$ that accepts at least one string. Observe that there exists a full-rank matrix $R \in \mathbb{F}_2^{n\times n}$ and a read-once regular BP f' of width $(r+1)$ such that for every $x \in \mathbb{F}_2^n$ we have $f(x) = f'(Rx)$. We claim that f' accepts at least one string in $H = \{Rx : x \in K\}$, which implies that f accepts at least one string in K.

For every $I \subseteq [n]$ of size r, observe that there exists $b \in \mathbb{F}_2^n$ such that $b + \mathrm{span}(\{R^{-1}e_i\}_{i\in I}) \subseteq K$, by definition of Kakeya set. This implies that $Rb + \mathrm{span}(\{e_i\}_{i\in I}) \subseteq H$. By Lemma 6.23, H is a hitting set for regular branching programs of width $(r+1)$. Therefore f' accepts at least one string in H. □

Corollary 6.24. *For every $r, n \in \mathbb{N}$ s.t. $r \le n$, there is an explicit hitting set $K \subseteq \mathbb{F}_2^n$ for oblivious read-once regular linear BP of width $(r+1)$ such that $|K| \le 2^{\lceil(1-2^{-r})n\rceil+r}$.*

6.4.1 Limitation to our approach

Next we prove Theorem 6.18, which proves a lower bound on rank-r Kakeya sets and implies that the seed length of hitting set generator based on our approach cannot be improved by much.

Theorem 6.25 (Theorem 6.18, restated)**.** *Every rank-r Kakeya set over $\mathbb{F}_2^n$ has size at least $2^{(1-2^{-r})(n+2)-r}$.*

Proof. Let $s_{n,r}$ denote the minimum size of rank-r Kakeya set over $\mathbb{F}_2^n$. Clearly $S_{n,0} = 1$ for every $n \in \mathbb{N}$. We will show that for every n, r we have $S_{n,r}^2 \geq 2^{n+1} S_{n-1,r-1}$, and then the claimed bound easily follows by induction.

To prove this claim, consider any rank-r Kakeya set over $\mathbb{F}_2^n$, denoted by K, and for every non-zero $a \in \mathbb{F}_2^n$ define $K_a = \{v \in \mathbb{F}_2^n : v \in K \wedge v + a \in K\}$. We claim that for every a we have $|K_a| \geq 2S_{n-1,r-1}$. (Note that this also implies $|K| \geq 2S_{n-1,r-1}$ because every K_a is a subset of K.) To prove this, first we assume w.l.o.g. that the n-th bit of a is 1, and define $K'_a = \{v' \in \mathbb{F}_2^{n-1} : v' \circ 0 \in K_a\}$. Note that $|K'_a| = |K_a|/2$ because for every $v \in \mathbb{F}_2^n$ we have $v \in K_a$ if and only if $v + a \in K_a$, and exactly one of $\{v, v + a\}$ has the last bit being 0.

We claim that K'_a is a rank-$(r-1)$ Kakeya set over size $\mathbb{F}_2^{n-1}$, and hence has size at least $S_{n-1,r-1}$. To prove this, consider any subspace $V' \subseteq \mathbb{F}_2^{n-1}$ of dimension $(r-1)$, and let V denote the subspace of $\mathbb{F}_2^n$ which consists of vectors in V' padded with a 0 in the last bit. Since K is a rank-r Kakeya set, there exists $b \in \mathbb{F}_2^n$ such that $b + V + \{0^n, a\} \subseteq K$. W.l.o.g. we can assume that the last bit of b is 0, i.e., $b = b' \circ 0$ for some $b' \in \mathbb{F}_2^{n-1}$. Then observe that $V' + b' \subseteq K'_a$, because for every $v' \in V'$ we have that $(v' \circ 0) + (b' \circ 0) \in K$ and $(v' \circ 0) + (b' \circ 0) + a \in K$, which implies that $v' + b' \in K'_a$.

Since the same argument works for every subspace V' of dimension $(r-1)$, this means K'_a is a rank-r Kakeya set. Finally, consider the bijective function $f : \mathbb{F}_2^n \times \mathbb{F}_2^n \to \mathbb{F}_2^n \times \mathbb{F}_2^n$ defined as $f(v_1, v_2) = (v_1, v_2 - v_1)$. Observe that the image of f on $K \times K$ is exactly $(K \times \{0^n\}) \cup \bigcup_{a \in \mathbb{F}_2^n, a \neq 0^n} K_a \times \{a\}$. This implies $|K|^2 \geq 2^{n+1} S_{n-1,r-1}$, which is exactly the bound we want.

□

Chapter 7

Weighted PRG for ROBPs

In this chapter, we present our contributions in the study of WPRGs for general read-once branching programs. Our first result is a construction of WPRG, which is the first simplified and improved construction over the original WPRG construction in [BCG20]. Although a later construction by Hoza [Hoz21a] achieved even better parameters than our result, in Chapter 8 we will see that our approach turns out to be quite useful in a specific setting called regular ROBPs. Our second contribution is an observation about how to apply WPRG in the Saks-Zhou derandomization framework. This observation was later used by Hoza [Hoz21a] in his proof of $\mathbf{BPL} \subseteq \mathbf{L}^{1.5-o(1)}$.

The results presented in this chapter is based on the following joint work:

- [CL20] Eshan Chattopadhyay and Jyun-Jie Liao. Optimal error pseudodistributions for read-once branching programs. In *35th Computational Complexity Conference, CCC 2020*, volume 169 of *LIPIcs*, pages 25:1–25:27. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020

7.1 Introduction

A natural approach to derandomize **BPL** is to construct explicit pseudorandom generators (PRGs) for read-once branching programs (ROBPs), which we formally define below. Note that the ROBPs here read the input bits in the standard order, i.e., from x_1 to x_n. This is different from what we saw in Chapter 6, where the queried bit can be adaptively chosen.

Definition 7.1 (ROBPs). *A ROBP B of length n and width w (or an (n, w)-*ROBP *for short) is specified by a start state $v_0 \in [w]$, a set of accept states V_{acc} and n transition functions $B_i : [w] \times \{0,1\} \to [w]$ for i from 1 to n. The ROBP B computes a function $B : \{0,1\}^n \to \{0,1\}$ as follows. Given an input $x \in \{0,1\}^n$, define $v_i = B_i(v_{i-1}, x_i)$, where x_i denotes the i-th bit of x. Then output $B(x) = 1$ if $v_n \in V_{\text{acc}}$, or $B(x) = 0$ otherwise.*

Remark 7.2. *Equivalently, one can view a ROBP B as a directed graph as follows. Consider $n+1$ layers of nodes $L_0, L_1, \ldots, L_n$, each having size w, and label the nodes in each L_i with $[w]$. For every $i \in [n], v \in [w], b \in \{0,1\}$, construct an edge with label b from v in L_{i-1} to $B_i(v, b)$ in L_i. Then the computation of $B(x)$ corresponds to a walk following label x from L_0 to L_n. In this paper we usually consider the equivalent graph view, and we refer to L_i as* layer i.

Definition 7.3 (PRGs). *Let $\mathcal{F}$ be a class of functions $f : \{0,1\}^n \to \{0,1\}$. An ε-PRG for $\mathcal{F}$ is a function $G : \{0,1\}^d \to \{0,1\}^n$ such that for every $f \in \mathcal{F}$,*

$$\left| \mathop{\mathbb{E}}_{x \sim \{0,1\}^n} [f(x)] - \mathop{\mathbb{E}}_{s \sim \{0,1\}^d} [f(G(s))] \right| \leq \varepsilon.$$

We say G ε-fools the class $\mathcal{F}$ if G is an ε-PRG for $\mathcal{F}$. We call d the seed length *of G. We say G is explicit if it can be computed in space $O(d)$.*[20]

[20] Throughout Chapter 7 and Chapter 8, when we say a generator G is explicit, it means G can be computed in space $O(d)$ where d is the input length of G.

It can be shown (via probabilistic method) that there exists a ε-PRG for width-w length-n ROBP with seed length $O(\log(nw/\varepsilon))$, which is optimal. Furthermore, an *explicit* PRG with such seed length would imply $\mathbf{BPL} = \mathbf{L}$. In a seminal work, Nisan [Nis92] constructed an explicit PRG with seed length $O(\log(n)\log(nw/\varepsilon))$, which is only a $O(\log(n))$ factor away from optimal. Nisan [Nis94] then used this PRG to prove that any problem in $\mathbf{BPL}$ can be deterministically computed in $O(\log^2(n))$ space and $\mathrm{poly}(n)$ time. Another remarkable work by Saks and Zhou [SZ99] also applied Nisan's generator in a non-trivial way to show that any problem in $\mathbf{BPL}$ can be deterministically computed in $O(\log^{3/2}(n))$ space (i.e., $\mathbf{BPL} \subseteq \mathbf{L}^{1.5}$).

7.1.1 Weighted PRGs

Despite decades of effort, the seed length of Nisan's PRG remains the state-of-the-art for width $w \geq 4$. In fact, even for the $w = 3$ case, Nisan's seed length remained unbeatable until a recent work by Meka, Reingold and Tal [MRT19] which improved the seed length to $\tilde{O}(\log(n)\log(1/\varepsilon))$. This has motivated researchers to study relaxed notions of PRGs and their applications in the derandomization of $\mathbf{BPL}$. A well-studied notion is that of a hitting set generator (HSG), which is the "one-sided" variant of a PRG.

Definition 7.4 (HSGs). *Let $\mathcal{F}$ be a class of functions $f : \{0,1\}^n \to \{0,1\}$. A ε-HSG for $\mathcal{F}$ is a function $G : \{0,1\}^d \to \{0,1\}^n$ such that for every $f \in \mathcal{F}$ s.t. $\mathbb{E}_{x\sim\{0,1\}^n}[f(x)] > \varepsilon$, it holds that $\mathbb{E}_{s\sim\{0,1\}^d}[f(G(s))] > 0$.*

The study of explicit HSGs for ROBPs has a long history, starting from the seminal work by Ajtai, Komlós and Szemerédi [AKS87]. While being weaker than PRGs,

explicit constructions of HSGs can still be used to derandomize randomized log-space algorithms with one-sided error (**RL**). In fact, a recent work by Cheng and Hoza [CH20] shows that an explicit HSG with optimal seed length $O(\log(nw/\varepsilon))$ already implies $\mathbf{BPL} = \mathbf{L}$.

In 2018, Braverman, Cohen and Garg [BCG20] introduced another relaxed notion of PRG called *weighted PRG* (WPRG). In this relaxed notion, each output string of G is further assigned a real weight that can *possibly be negative*.

Definition 7.5. *Let $\mathcal{F}$ be a class of functions $f : \{0,1\}^n \to \{0,1\}$. A ε-WPRG is a pair of functions $(\rho, G) : \{0,1\}^d \to \{0,1\}^n \times \mathbb{R}$ such that for every $f \in \mathcal{F}$,*

$$\left| \mathop{\mathbb{E}}_{x \sim \{0,1\}^n} [f(x)] - \mathop{\mathbb{E}}_{s \sim \{0,1\}^d} [\rho(s) \cdot f(G(s))] \right| \leq \varepsilon.$$

We say (ρ, G) is K-bounded if $|\rho(s)| \leq K$ for every $s \in \{0,1\}^d$.

It was observed in [BCG20] that ε-WPRGs implies ε-HSGs. In addition, WPRGs seem closer to PRGs than HSGs in the sense that one can use a WPRG to (two-sidedly) estimate the expectation of a ROBP f by simply enumerating all the seeds.

Surprisingly, by simply allowing negative weights, Braverman, Cohen and Garg [BCG20] showed how to construct an explicit ε-WPRG with seed length $\tilde{O}(\log(n)\log(nw)+\log(1/\varepsilon))$. This improves over the seed length of Nisan's generator when the error is small ($\varepsilon < n^{-\Omega(\log(nw)\log\log(nw))}$) and achieves near optimal dependence on the error parameter ε. However, a drawback of the result in [BCG20] is that their construction and analysis are complicated. Subsequent to [BCG20], Hoza and Zuckerman [HZ20] showed a simple construction of an explicit HSG with seed length $O(\log(n)\log(nw)+\log(1/\varepsilon))$. However, their techniques do not seem to work for WPRGs.

7.1.2 Our Results

The main result in this chapter is an explicit WPRG with optimal dependence on error (up to a constant factor).

Theorem 7.6. *There exists an explicit ε-WPRG (ρ, G) for (n, w)-ROBPs with seed length*

$$O\left(\log(n)\log(nw)\cdot\log\log(nw)+\log(1/\varepsilon)\right),$$

which is $\mathrm{poly}(1/\varepsilon)$-bounded.

This improves upon the construction in [BCG20] by a factor of $O(\log\log(1/\varepsilon))$, for any $\varepsilon < n^{-\Omega(\log(nw)\log\log(nw))}$. In addition, our construction and analysis are arguably simpler than those in [BCG20].

We note that similar to the results in subsequent works [CDR+21, PV21, Hoz21a], our construction can also be viewed as a black-box error reduction procedure. This might not be obvious in our original analysis in [CL20], so we give a new exposition in this chapter and formally prove the following theorem.

Theorem 7.7. *Suppose there exists an explicit (n^{-3})-PRG for (n, w)-ROBPs with seed length d_0. Then for every $\varepsilon > 0$ there exists an explicit $\mathrm{poly}(1/\varepsilon)$-bounded ε-WPRG with seed length*

$$d = d_0 + O(\log(n)\log(nw)\log\log_n(1/\varepsilon)+\log(1/\varepsilon)).$$

Note that this reduction together with Nisan's PRG [Nis92] ($d_0 = O(\log(n)\log(nw))$) imply Theorem 7.6.[21]

As observed in [BCG20], the small-error regime is well motivated for application to derandomizing space-bounded computation. In particular, Saks and Zhou [SZ99]

[21] The $\log\log_n(1/\varepsilon)$ factor can be replaced with $\log\log(nw)$ because the $O(\log(1/\varepsilon))$ term would be dominating if $\log\log_n(1/\varepsilon) = \omega(\log\log(nw))$.

instantiated Nisan's PRG with error $n^{-\omega(1)}$ to prove that $\mathbf{BPL} \subseteq \mathbf{L}^{3/2}$. In fact, in [BCG20] it was conjectured that "having some further guarantees on the ρ might allow one to apply the Saks-Zhou schemes with WPRGs as well," and in Section 7.6 we give an affirmative answer: one can apply the Saks-Zhou scheme on a WPRG as long as it's $\mathrm{poly}(w, 1/\epsilon)$-bounded.

Our construction uses a strategy similar to [BCG20] with the following key differences.

- The construction in [BCG20] has a more *bottom-up* nature: their construction follows the binary-tree structure in Nisan's PRG [Nis92], but in each node they maintain a sophisticated "leveled matrix representation" (LMR) which consists of many pieces of small-norm matrices that are generated from a telescoping-sum trick. They then showed how to combine pieces in two LMRs one by one to form a LMR in the upper level. On the other hand, our construction also follows the binary-tree structure, but has a more *top-down* nature: we give a clean recursive formula which generates a WPRG for length-n ROBPs given WPRGs for shorter ROBPs with larger error. The top-down nature significantly simplifies the construction and analysis because we no longer care about the implementation details in lower levels.
- A key observation in [BCG20] which makes their implementation work is the use of *averaging samplers*. In more details, they utilized the fact that *failure probability* is much cheaper than *approximation error* in samplers. To implement this idea, [BCG20] made a distinction between two different parts of the seed, an outer seed and an inner seed. Roughly speaking, they needed to maintain the property that with high probability over the outer seed, the average over the inner seeds gives a good approximation. However, the inner seed could grow

faster than it should be if not treated carefully, so [BCG20] carefully applied different multiplication rules for different cases to make sure that the inner seed does not grow too fast. In our construction, we observe that the distinction between inner seed and outer seed is no longer necessary once a sampler is applied. Therefore we do not need any special-case treatment. The improvement in our seed length also comes from this observation.

Subsequent Works. Pyne and Vadhan [PV21] and Cohen, Doron, Renard, Sberlo and Ta-Shma [CDR+21] discovered a different WPRG construction with seed length $O(\log(n)\log(nw)+\log(1/\varepsilon)\log\log(1/\varepsilon))$. Their construction is based on the "inverse-Laplacian" perspective of ROBPs and "Richardson iteration", which was first introduced to the space-bounded derandomization context by Ahmadinejad, Kelner, Murtagh, Peebles, Sidford and Vadhan [AKM+20]. Their seed length is incomparable to ours, and the difference in the seed length comes from their different choice of derandomization technique. In this work we use averaging samplers in a somewhat similar (but simpler) way to [BCG20]. In [PV21, CDR+21], they apply the INW generator [INW94] (a variant of Nisan's PRG) on ROBPs with larger step size instead. We will see their derandomization tricks in more details in Chapter 8.

Subsequent to these two works, Hoza [Hoz21a] observed that Richardson iteration together with averaging samplers imply a WPRG with improved seed length $O(\log(n)\log(nw)+\log(1/\varepsilon))$. In the same work Hoza also proved the surprising result that $\mathbf{BPL}\subseteq\mathbf{L}^{1.5-o(1)}$, using the WPRG construction in [PV21, CDR+21] (as a black-box error reduction), an improved PRG construction for $w\gg n$ by Armoni [Arm98] and our argument about the Saks-Zhou scheme on WPRGs in Section 7.6.

7.2 ROBPs and Matrices

Before we discuss our construction, first we introduce the equivalence between ROBPs and matrices. Consider a ROBP B of length n and width w specified by transition functions $B_1, \ldots, B_n$. For every $i \in [n]$, and every $b \in \{0,1\}$, define the matrix $\mathbf{M}_i(b) \in \mathbb{R}^{w \times w}$ as

$$\forall u, v \in [w], \mathbf{M}_i(b)[u,v] := \begin{cases} 1 \text{ if } B_i(u,b) = v, \\ 0 \text{ otherwise.} \end{cases}$$

We refer to $\mathbf{M}_i(b)$ as the *transition matrix of* B to layer i on input b. In addition, we denote the transition matrices $(\mathbf{M}_1(0), \mathbf{M}_1(1), \ldots, \mathbf{M}_n(0), \mathbf{M}_n(1))$ collectively with $\mathbf{M}_{[n]}$ for short. Furthermore, for every $0 \le \ell < r \le n$ and a string $s \in \{0,1\}^{r-\ell}$, we denote the *transition matrix mapping* from layer ℓ to layer r as

$$\mathbf{M}_{\ell..r}(s) := \prod_{i=1}^{r-\ell} \mathbf{M}_{\ell+i}(s_i).$$

If we do not specify ℓ, r, we implicitly assume $\ell = 0$ and $r = n$ by default. The following fact is straightforward by definition.

Fact 7.8. *For every $\ell < m < r$ and $x \in \{0,1\}^{m-\ell}, y \in \{0,1\}^{r-m}$, $\mathbf{M}_{\ell..m}(x)\mathbf{M}_{m..r}(y) = \mathbf{M}_{\ell..r}(x \circ y)$.*

In addition, note that $\mathbf{M}_{\ell..r}(s)$ is also the transition matrix on input s of the $(r-\ell, w)$-ROBP specified by transition functions $B_{\ell+1}, \ldots, B_r$ (which we call the *subprogram from layer ℓ to layer r*). Finally, we abuse notation and further define $\mathbf{M}_i := \frac{1}{2}(\mathbf{M}_i(0) + \mathbf{M}_i(1))$ which we call the *random-walk matrix* of B_i. Similarly, for every ℓ, r we define $\mathbf{M}_{\ell..r} := \mathbb{E}_{s \sim \mathbf{U}_{r-\ell}}[\mathbf{M}_{\ell..r}(s)] = \prod_{i=\ell+1}^{r} \mathbf{M}_i$ which is the *random-walk matrix from layer ℓ to layer r*.

Now we are ready to discuss the equivalence between constructing WPRGs and approximating the random-walk matrix from layer 0 to layer n. First of all, observe that for a start state $v_0 \in [w]$ and a set of accept states $V_{\text{acc}} \subseteq [w]$, $B(x) = 1$ if and only if there exists $v_n \in V_{\text{acc}}$ s.t. $\mathbf{M}_{0..n}(x)[v_0, v_n] = 1$. Equivalently, if we define v_{st} to be the "start vector" s.t. $v_{\mathsf{st}}[v_0] = 1$ and $v_{\mathsf{st}}[i] = 0$ for every $i \neq v_0$, and v_{ed} to be the "accept vector" s.t. $v_{\mathsf{ed}}[i] = 1$ if $i \in V_{\text{acc}}$ and $v_{\mathsf{ed}}[i] = 0$ otherwise, then $B(x) = 1$ if and only if $v_{\mathsf{st}}^\top \mathbf{M}_{0..n}(x) v_{\mathsf{ed}} = 1$. Then observe that for any pair of functions $(\rho, G) : \{0,1\}^d \to \mathbb{R} \times \{0,1\}^n$,

$$\begin{aligned} &\left| \mathop{\mathbb{E}}_{s \sim \mathbf{U}_d} [\rho(s) B(G(s))] - \mathop{\mathbb{E}}_{x \sim \mathbf{U}_n} [B(x)] \right| \leq \varepsilon \\ \Leftrightarrow &\left| v_{\mathsf{st}}^\top \left(\mathop{\mathbb{E}}_{s \sim \mathbf{U}_d} [\rho(s) \mathbf{M}_{0..n}(G(s))] - \mathbf{M}_{0..n} \right) v_{\mathsf{ed}} \right| \leq \varepsilon. \end{aligned}$$

Because $\|v_{\mathsf{ed}}\|_\infty \leq 1$ and $\|v_{\mathsf{st}}\|_1 \leq 1$, to show that (ρ, G) is a ε-WPRG, it suffices to show that

$$\left\| \mathop{\mathbb{E}}_{s \sim \mathbf{U}_d} [\rho(s) \mathbf{M}_{0..n}(G(s))] - \mathbf{M}_{0..n} \right\|_\infty \leq \varepsilon.$$

In fact, because for (ρ, G) to be a ε-WPRG we need the inequalities above to work for every choice of $v_0 \in [w]$ and $V_{\text{acc}} \subseteq [w]$, a bound on matrix infinity norm is not only sufficient but also necessary. We formalize this claim in the following lemma.

Lemma 7.9. *If (ρ, G) is a ε-WPRG for (n, w)-ROBPs with seed length d, then for every (n, w)-ROBP and its corresponding transition matrix mapping $\mathbf{M}_{0..n}(\cdot)$,*

$$\left\| \mathop{\mathbb{E}}_{s \sim \mathbf{U}_d} [\rho(s) \mathbf{M}_{0..n}(G(s))] - \mathbf{M}_{0..n} \right\|_\infty \leq 2\varepsilon.$$

Proof. It suffices to prove that for every $y \in \mathbb{R}^w$ s.t. $\|y\|_\infty \leq 1$,

$$\left\| \left(\mathop{\mathbb{E}}_{s \sim \mathbf{U}_d} [\rho(s) \mathbf{M}_{0..n}(G(s))] - \mathbf{M}_{0..n} \right) y \right\|_\infty \leq 2\varepsilon.$$

Observe that for every $i \in [w]$, because $-1 \leq y[i] \leq 1$, we have $y[i] = \int_0^1 \mathbb{1}(y[i] \geq t)\, dt - \int_0^1 \mathbb{1}(y[i] \leq -t)\, dt$. Therefore, if we let $y_t^+ \in (\{0,1\})^w$ denote the vector that

$y_t^+[i] = \mathbb{1}(y[i] \geq t)$ and y_t^- denote the vector that $y_t^+[i] = \mathbb{1}(y[i] \leq -t)$ for every $t \in [0,1]$, we have $y = \int_0^1 (y_t^+ - y_t^-)\, dt$. Furthermore, for every y_t^+ and every $i \in [w]$, if we consider the ROBP with transition matrices $\mathbf{M}_{0..n}(\cdot)$, start state i and end states $\{i \in [w] : y_i \geq t\}$, by the property of WPRG we have that the i-th coordinate of $(\mathbb{E}_{s\sim\mathbf{U}_d}[\rho(s)\mathbf{M}_{0..n}(G(s))] - \mathbf{M}_{0..n})\, y_t^+$ has absolute value at most ε. Similarly, the i-th coordinate of $(\mathbb{E}_{s\sim\mathbf{U}_d}[\rho(s)\mathbf{M}_{0..n}(G(s))] - \mathbf{M}_{0..n})\, y_t^-$ has absolute value at most ε. Therefore, the i-th coordinate of $(\mathbb{E}_{s\sim\mathbf{U}_d}[\rho(s)\mathbf{M}_{0..n}(G(s))] - \mathbf{M}_{0..n})\, y$ has absolute value

$$\left| \int_0^1 \left(\mathop{\mathbb{E}}_{s\sim\mathbf{U}_d} [\rho(s)\mathbf{M}_{0..n}(G(s))] - \mathbf{M}_{0..n} \right) (y_t^+ - y_t^-)\, dt \right|,$$

which is at most 2ε by triangle inequality. □

Based on the equivalence above, we mainly focus on constructing a matrix $\widetilde{\mathbf{M}_{0..n}}$ such that $\left\| \widetilde{\mathbf{M}_{0..n}} - \mathbf{M}_{0..n} \right\|_\infty \leq \varepsilon$, but at the same time we make sure that there is a pair of functions (ρ, G) such that $\widetilde{\mathbf{M}_{0..n}} = \mathbb{E}_{s\sim\mathbf{U}_d}[\rho(s)\mathbf{M}_{0..n}(G(s))]$, which will be our WPRG. To avoid some technical caveats, we consider a "natural form" of WPRGs which can take non-binary input. First we define a realization of a matrix.

Definition 7.10. *Let $\mathbf{M} : \{0,1\}^n \to \mathbb{R}^{w\times w}$ be the transition matrix mapping of a (n,w)-ROBP. We say $\rho : [K] \to \{\pm 1, 0\}$ and $G : [K] \times \{0,1\}^d \to \{0,1\}^n$ form a (K,d)-*realization *of a matrix $\widetilde{\mathbf{M}}$ based on $\mathbf{M}$ if*

$$\widetilde{\mathbf{M}} = \sum_{i\in[K]} \rho(i) \mathop{\mathbb{E}}_{r\sim\mathbf{U}_d} [\mathbf{M}(G(i,r))]\,.$$

We say K is the boundedness *and d is the* seed length *of (ρ, G).*

Throughout this chapter, we sometimes do not specify the transition matrices when they do not matter or are clear from the context. Specifically, we use the convention that a realization of a matrix denoted by $\mathbf{M}_{\ell..r}$ with additional superscripts/subscripts (e.g. $\widetilde{\mathbf{M}_{\ell..r}}$, $\mathbf{M}_{\ell..r}^{(k)}$ etc.) is based on the subprogram from layer ℓ to r

and its transition matrices $\mathbf{M}_{\ell..r}(\cdot)$. Furthermore, we note that any (K, d)-realization can be turned into a (K', d')-realization for any $K' \geq K$ and $d' \geq d$, if we simply define $\rho(i) = 0$ for any $i > K$ and ignore the last $(d' - d)$ bits of the second input of G. Therefore, to prove that some matrix $\widetilde{\mathbf{M}_{i..j}}$ has a (K', d')-realization, it suffices to prove that it has a (K, d)-realization for some $K \leq K'$ and $d \leq d'$.

Next we define natural WPRGs.

Definition 7.11. *We say a pair of functions (ρ, G) is a (K, d, ε)-natural WPRG for (n, w)-ROBPs if for every (n, w)-ROBP and its corresponding transition matrix mapping $\mathbf{M}_{0..n}(\cdot)$, (ρ, G) is a (K, d)-realization of a matrix $\widetilde{\mathbf{M}_{0..n}}$ such that*

$$\left\| \widetilde{\mathbf{M}_{0..n}} - \mathbf{M}_{0..n} \right\|_\infty \leq \varepsilon.$$

We say (ρ, G) is explicit if (ρ, G) is computable in space $O(d + \log(K))$.

It's easy to see that a natural WPRG implies a WPRG with standard definition.

Lemma 7.12. *If there is an explicit (K, d, ε)-natural WPRG (ρ, G) for (n, w)-ROBPs, then there is also an explicit ε-WPRG (ρ', G') for (n, w)-ROBPs with seed length $d + \lceil \log K \rceil$ that is $2K$-bounded.*

Proof. We define $(\rho', G') : \{0,1\}^{d+\lceil \log K \rceil} \to \mathbb{R} \times \{0,1\}^n$ as follows. Given input $r' \in \{0,1\}^{d+\lceil \log K \rceil}$, first we parse it as a pair $(i, r) \in \{0,1\}^{\lceil \log K \rceil} \times \{0,1\}^r$. Then we interpret i as an integer in $[2^{\lceil \log K \rceil}]$. If $i \leq K$, then define $\rho'(r') = 2^{\lceil \log K \rceil} \cdot \rho(i)$ and $G'(r') = G(i, r)$. Otherwise, define $\rho'(r') = 0$ and $G'(r')$ as an arbitrary n-bit string. Observe that

$$\mathop{\mathbb{E}}_{r' \sim \mathbf{U}_{d+\lceil \log K \rceil}} \left[\rho'(r') \mathbf{M}_{0..n}(G'(r')) \right] = \sum_{i \in [K]} \rho(i) \mathop{\mathbb{E}}_{r \sim \mathbf{U}_d} \left[\mathbf{M}_{0..n}(G(i, r)) \right].$$

Because (ρ, G) realizes a matrix that is ε-close to $\mathbf{M}_{0..n}$ in $\|\cdot\|_\infty$, we can conclude that

$$\left\| \mathop{\mathbb{E}}_{r' \sim \mathbf{U}_{d+\lceil \log K \rceil}} \left[\rho'(r') \mathbf{M}_{0..n}(G'(r')) \right] - \mathbf{M}_{0..n} \right\|_\infty \leq \varepsilon,$$

Which implies that (ρ', G') is a ε-WPRG. For the boundedness, note that for every r', $|\rho'(r')| \leq 2^{\lceil \log K \rceil} \leq 2K$. □

Finally we briefly discuss how to translate arithmetic operations of matrices to their realizations.

Lemma 7.13. *If there is a (K_1, d_1)-realization (ρ_1, G_1) of $\widetilde{\mathbf{M}_{\ell..m}}$ and a (K_2, d_2)-realization (ρ_2, G_2) of $\widetilde{\mathbf{M}_{m..r}}$, then there is also a $(K_1K_2, d_1 + d_2)$-realization of $\widetilde{\mathbf{M}_{\ell..m}}\widetilde{\mathbf{M}_{m..r}}$.*

Proof. Given $i \in [K_1K_2]$ and $s \in \{0,1\}^{d_1+d_2}$, define $i_1 = \lceil i/K_2 \rceil$, $i_2 = i - (i_1 - 1) \cdot K_2$, and let s_1 be the first d_1 bits of s and s_2 be the last d_2 bits of s. Then define $\rho(i) = \rho_1(i_1)\rho_2(i_2)$ and $G(i,s) = G_1(i_1, s_1) \circ G_2(i_2, s_2)$. To prove the correctness of (ρ, G), first observe that by Fact 7.8 we have $\mathbf{M}_{\ell..r}(G(i,s)) = \mathbf{M}_{\ell..m}(G_1(i_1,s_1))\mathbf{M}_{m..r}(G_2(i_2,s_2))$. In addition, note that the mappings $i \to (i_1, i_2)$ and $s \to (s_1, s_2)$ are both bijective. Therefore by distributive law we have

$$\begin{aligned}
&\widetilde{\mathbf{M}_{\ell..m}}\widetilde{\mathbf{M}_{m..r}} \\
&= \left(\sum_{i_1=1}^{K_1} \rho_1(i_1) \mathop{\mathbb{E}}_{s_1 \sim \mathbf{U}_{d_1}} [\mathbf{M}_{\ell..m}(G_1(i_1,s_1))] \right) \left(\sum_{i_2=1}^{K_2} \rho_2(i_2) \mathop{\mathbb{E}}_{s_2 \sim \mathbf{U}_{d_2}} [\mathbf{M}_{m..r}(G_2(i_2,s_2))] \right) \\
&= \sum_{i \in [K_1K_2]} \rho(i) \mathop{\mathbb{E}}_{s \sim \mathbf{U}_{d_1+d_2}} [\mathbf{M}_{\ell..r}(G(i,s))] .
\end{aligned}$$

□

Lemma 7.14. *If there is a (K, d)-realization (ρ, G) of $\widetilde{\mathbf{M}}$, then there is also a (K, d)-realization of $-\widetilde{\mathbf{M}}$.*

Proof. Simply negate the output of ρ. □

Lemma 7.15. *If for each $j \in [t]$ there is a (K_j, d)-realization (ρ_j, G_j) of $\mathbf{M}_{(j)}$ based on the transition matrix mapping $\mathbf{M}$, then $\sum_{j\in[t]} \mathbf{M}_{(j)}$ also has a $(\sum_{j\in[t]} K_j, d)$-realization based on $\mathbf{M}$.*

Proof. Given $i \in [\sum_j K_j]$ and $s \in \{0,1\}^d$, let $j^* = \arg\min_r(\sum_{j=1}^{r} K_r \geq i)$, and $i^* = i - \sum_{j=1}^{j^*-1} K_j$. Then define $\rho(i) = \rho_{j^*}(i^*)$ and $G(i,s) = G_{j^*}(s)$. To prove the correctness, observe that the mapping $i \to (j^*, i^*)$ (from $[\sum_j K_j]$ to $\bigcup_{j\in[t]}\{j\} \times [K_j]$) is bijective. Therefore

$$\begin{aligned}\sum_i \rho(i) \mathop{\mathbb{E}}_{s\sim\mathbf{U}_d}[\mathbf{M}(G(i,s))] &= \sum_{j^*\in[t]} \sum_{i^*\in[K_{j^*}]} \rho_{j^*}(i^*) \mathop{\mathbb{E}}_{s\sim\mathbf{U}_d}[\mathbf{M}(G_{j^*}(i^*,s))] \\ &= \sum_{j\in[t]} \mathbf{M}_{(j)}.\end{aligned}$$

□

7.3 Recursive Formula

As claimed in Section 7.1.2, the simplicity of our construction comes from its top-down nature. In this section, we show the matrix identity which our top-down construction is based on, and briefly discuss how it is used in our final WPRG construction.

Lemma 7.16 ([CL20])**.** *Consider $w\times w$ matrices $\mathbf{A}, \mathbf{A}^{(0)}, \mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(k)}$ and $\mathbf{B}, \mathbf{B}^{(0)}, \ldots, \mathbf{B}^{(k)}$, and define*

$$(\mathbf{AB})^{(k)} := \sum_{i+j=k} \mathbf{A}_i\mathbf{B}_j - \sum_{i+j=k-1} \mathbf{A}_i\mathbf{B}_j.^{22}$$

Then

$$(\mathbf{AB})^{(k)} - \mathbf{AB} = \sum_{i+j=k} \mathbf{\Delta}_A^{(i)}\mathbf{\Delta}_B^{(j)} - \sum_{i+j=k-1} \mathbf{\Delta}_A^{(i)}\mathbf{\Delta}_B^{(j)} + \mathbf{\Delta}_A^{(k)}\mathbf{B} + \mathbf{A}\mathbf{\Delta}_B^{(k)}.$$

where $\mathbf{\Delta}_A^{(i)} := \mathbf{A}^{(i)} - \mathbf{A}$ and $\mathbf{\Delta}_B^{(i)} := \mathbf{B}^{(i)} - \mathbf{B}$ for $i \in \{0,1,\ldots,k\}$.

[22] In the summations we eunmerate over all non-negative integers i, j that satisfy the given constraint. We use similar convention throughout Chapter 7 and Chapter 8.

The proof of this identity is straightforward by distributive law of matrix multiplication. To see why this formula would be useful, next we show a toy construction that builds a matrix $\mathbf{M}^{(k)}_{0..n}$ which is a $(n\varepsilon_0)^k$-approximation of $\mathbf{M}_{0..n}$ from matrices $\mathbf{M}^{(0)}_{\ell..r}$ that only ε_0-approximate $\mathbf{M}_{\ell..r}$. Based on the equivalence between WPRGs and matrices that we discussed in the previous section, given PRGs with error $\varepsilon_0 < 1/\operatorname{poly}(n)$ for (n, w)-ROBPs we can get WPRGs with arbitrarily small error for (n, w)-ROBPs. Note that this construction itself does not give a WPRG with good enough seed length, but our final WPRG construction is just a proper derandomization of this construction. Without loss of generality, assume that n is a power of 2.[23] In addition, for ease of notation we define the set of pairs

$$\mathsf{BS}_n = \{(\ell, r) : \exists i, k \in \mathbb{N} \cup \{0\} \text{ s.t. } \ell = i \cdot 2^k, r = \ell + 2^k \text{ and } 0 \le \ell < r \le n\},$$

which denotes the indices of all the transition matrix mappings $\mathbf{M}_{\ell..r}$ that we need to consider in our recursive construction. Note that $(0, n) \in \mathsf{BS}_n$, In addition, for any $(\ell, r) \in \mathsf{BS}_n$ such that $r - \ell > 1$, both (ℓ, m) and (m, r) are also in BS_n, where $m = (\ell + r)/2$.

Lemma 7.17. *Consider a (n, w)-ROBP and its transition matrices $\mathbf{M}_{[n]}$ as defined in Section 7.2. Fix an error parameter $\varepsilon_0 < 1/4n^2$. For every $(\ell, r) \in \mathsf{BS}_n$, let $\mathbf{M}^{(0)}_{\ell..r}$ be a matrix such that $\left\|\mathbf{M}^{(0)}_{\ell..r} - \mathbf{M}_{\ell..r}\right\|_\infty \le \varepsilon_0$. For every $k \in \mathbb{N}$ and every $(\ell, r) \in \mathsf{BS}_n$, recursively define*

$$\mathbf{M}^{(k)}_{\ell..r} := \begin{cases} \mathbf{M}_r & \text{if } r - \ell = 1, \\ \sum_{i+j=k} \mathbf{M}^{(i)}_{\ell..m} \cdot \mathbf{M}^{(j)}_{m..r} - \sum_{i+j=k-1} \mathbf{M}^{(i)}_{\ell..m} \cdot \mathbf{M}^{(j)}_{m..r} & \text{otherwise}, \end{cases}$$

where $m = (\ell + r)/2$. Then for every $k \in \mathbb{N}$

$$\left\|\mathbf{M}^{(k)}_{0..n} - \mathbf{M}_{0..n}\right\|_\infty \le \binom{2n-1}{k+1} \varepsilon_0^{k+1}.$$

[23] Note that any ROBP of length n can be turned into another ROBP of longer length n' by adding $(n' - n)$ identity transitions. Therefore to get a WPRG for (n, w)-ROBPs we can simply take a WPRG for (n', w)-ROBPs (ρ, G) and truncate the last $(n' - n)$ bits of the output of G.

Proof. We prove by induction on $(r-\ell), k$ that

$$\left\|\mathbf{M}^{(k)}_{\ell..r} - \mathbf{M}_{\ell..r}\right\|_\infty \le \binom{2(r-\ell)-1}{k+1}\varepsilon_0^{k+1}.$$

The base cases $r-\ell = 1$ is trivial because $\left\|\mathbf{M}^{(k)}_{\ell..r} - \mathbf{M}_{\ell..r}\right\|_\infty = 0$. The $k = 0$ and $r-\ell > 1$ case is also straightforward by definition. For the other cases, first we let $\mathbf{\Delta}^{(t)}_{i..j}$ denote $\mathbf{M}^{(t)}_{i..j} - \mathbf{M}_{i..j}$ for every i, j, t. Then we expand $\mathbf{M}^{(k)}_{\ell..r} - \mathbf{M}_{\ell..r}$ by Lemma 7.16 and apply sub-multiplicativity and sub-additivity of $\|\cdot\|_\infty$. This will give us

$$\left\|\mathbf{\Delta}^{(k)}_{\ell..r}\right\|_\infty \le \sum_{i+j\in\{k-1,k\}} \left\|\mathbf{\Delta}^{(i)}_{\ell..m}\right\|_\infty \left\|\mathbf{\Delta}^{(j)}_{m..r}\right\|_\infty + \left\|\mathbf{\Delta}^{(k)}_{\ell..m}\right\|_\infty + \left\|\mathbf{\Delta}^{(k)}_{m..r}\right\|_\infty.$$

Next, by induction hypothesis, the above inequality imply

$$\left\|\mathbf{\Delta}^{(k)}_{\ell..r}\right\|_\infty \le C_1\varepsilon_0^{k+1} + C_2\varepsilon_0^{k+2},$$

where

$$C_1 = \sum_{i+j=k-1} \binom{2(m-\ell)-1}{i+1}\binom{2(r-m)-1}{j+1} + \binom{2(m-\ell)-1}{k+1} + \binom{2(r-m)-1}{k+1}$$

and

$$C_2 = \sum_{i+j=k} \binom{2(m-\ell)-1}{i+1}\binom{2(r-m)-1}{j+1}.$$

Observe that $C_1 = \binom{2(r-\ell)-2}{k+1}$ by Vandermonde's identity, and similarly $C_2 \le \binom{2(r-\ell)-2}{k+2} \le 4n^2\binom{2(r-\ell)-2}{k}$. Given $\varepsilon_0 \le 1/4n^2$, this implies $C_1 + C_2\varepsilon_0 \le \binom{2(r-\ell)-2}{k+1} + \binom{2(r-\ell)-2}{k+2} = \binom{2(r-\ell)-1}{k+2}$. Therefore

$$\left\|\mathbf{\Delta}^{(k)}_{\ell..r}\right\|_\infty \le (C_1 + C_2\varepsilon_0)\varepsilon_0^{k+1} \le \binom{2(r-\ell)-1}{k+1}\varepsilon_0^{k+1}.$$

□

7.4 WPRG Construction

Next we show our actual WPRG construction. Briefly speaking, to compute $\mathbf{M}^{(k)}_{0..n}$ with short seed length, we apply the same recursion as in Lemma 7.17, with the only

difference being that we take the base cases to be $\mathbf{M}^{(h)}_{\ell..r}$ for every $h \le \lfloor k/2 \rfloor$ instead of $\mathbf{M}^{(0)}_{\ell..r}$. These base cases $\mathbf{M}^{(h)}_{\ell..r}$ are also computed recursively with the same recursive formula, except that we further use *averaging samplers* to recycle the randomness that is used in each base-case matrix. Then we repeat this process for $\lceil \log k \rceil$ times until we eventually reduce the task to computing $\mathbf{M}^{(0)}_{\ell..r}$, which can be done using a base-case PRG with larger error. We discuss the details below. (The space complexity analysis is deferred to Section 7.5.)

Again we assume without loss of generality that n is a power of 2. First we prove the following lemma that analyzes the error and realization cost of $\mathbf{M}^{(k)}_{0..n}$, when the base cases are taken to be $\mathbf{M}^{(h)}_{\ell..r}$ for every $h \le \lfloor k/2 \rfloor$.

Lemma 7.18. *Consider a (n, w)-ROBP and its transition matrices $\mathbf{M}_{[n]}$ as defined in Section 7.2. Fix an error parameter $\varepsilon_0 < 1/4n^2$ and $k, A, B \in \mathbb{N}$. Suppose that for every integers ℓ, r, h s.t. $(\ell, r) \in \mathsf{BS}_n$ and $h \le \lfloor k/2 \rfloor$, the matrix $\mathbf{M}^{(h)}_{\ell..r}$ is already defined. For every $\lfloor k/2 \rfloor < h \le k$ and every ℓ, r s.t. $(\ell, r) \in \mathsf{BS}_n$, recursively define*

$$\mathbf{M}^{(h)}_{\ell..r} := \begin{cases} \mathbf{M}_r & \text{if } r - \ell = 1, \\ \sum_{i+j=h} \mathbf{M}^{(i)}_{\ell..m} \cdot \mathbf{M}^{(j)}_{m..r} - \sum_{i+j=h-1} \mathbf{M}^{(i)}_{\ell..m} \cdot \mathbf{M}^{(j)}_{m..r} & \text{otherwise,} \end{cases}$$

where $m = (\ell + r)/2$. Then the following holds.

- *If every base case $\mathbf{M}^{(h)}_{\ell..r}$ satisfies that $\left\| \mathbf{M}^{(h)}_{\ell..r} - \mathbf{M}_{\ell..r} \right\|_\infty \le \binom{2(r-\ell)-1}{h+1} \varepsilon_0^{h+1}$, then $\left\| \mathbf{M}^{(k)}_{0..n} - \mathbf{M}_{0..n} \right\|_\infty \le \binom{2n-1}{k+1} \varepsilon_0^{k+1}$.*
- *If every base case $\mathbf{M}^{(h)}_{\ell..r}$ has a $(\binom{2(r-\ell)+h-2}{h}, hA + B)$-realization $(\rho^{(k)}_{\ell..r}, G^{(k)}_{\ell..r})$ for some $A, B \in \mathbb{N}$, then $\mathbf{M}^{(k)}_{0..n}$ has a $(\binom{2n+k-2}{k}, kA + \log(n)B)$-realization (ρ, G). In addition, ρ depends only on $\rho^{(k)}_{\ell..r}$ but not on $G^{(k)}_{\ell..r}$*

Proof. The first bullet point follows from the exact same inductive proof for Lemma 7.17. To prove the second bullet point, we prove by induction on $(r - \ell), h$

that for every ℓ, r, h s.t. $(\ell, r) \in \mathsf{BS}_n$ and $h \le k$, $\mathbf{M}^{(h)}_{\ell..r}$ has a $(\binom{2(r-\ell)+h-2}{h}, hA + \log(r-\ell)B)$-realization. The base case $h \le \lfloor k/2 \rfloor$ is trivial by definition. For the other base case $r - \ell = 1$, note that $\mathbf{M}_r$ has a $(1,1)$-realization, which also satisfies the inductive hypothesis give that $B \ge 1$. Next we proceed to the inductive case, and consider

$$\mathbf{M}^{(h)}_{\ell..r} := \sum_{i+j=h} \mathbf{M}^{(i)}_{\ell..m} \cdot \mathbf{M}^{(j)}_{m..r} - \sum_{i+j=h-1} \mathbf{M}^{(i)}_{\ell..m} \cdot \mathbf{M}^{(j)}_{m..r}.$$

By Lemma 7.13, Lemma 7.14 and Lemma 7.15 there exists a (K, d)-realization of $\mathbf{M}^{(h)}_{\ell..r}$, where K and d are bounded as follows. To bound K, first observe that

$$K \le \sum_{i+j \in \{h-1,h\}} \binom{2(m-\ell)+i-2}{i} \binom{2(r-m)+j-2}{j}.$$

Then recall the basic combinatorial fact that $\binom{a+b-1}{b}$ counts the number of length-a sequences of non-negative integers that sum up to b. We claim that the right hand side of the inequality above partially counts the number of length-$(2(r-\ell)-1)$ sequences that sum up to h, and thus is upperbounded by $\binom{2(r-\ell)+h-2}{h}$. To see why this is true, first observe that each $i+j=h$ term counts the number of sequences that starts with a 0, follows by a length-$(2(m-\ell)-1)$ sequence that sums up to i and then a length-$(2(r-m)-1)$ sequence that sums up to j. Similarly, each $i+j=h-1$ term counts the number of sequences that starts with a 1, follows by a length-$(2(m-\ell)-1)$ sequence that sums up to i and then a length-$(2(r-m)-1)$ sequence that sums up to j. Each of the sequence that gets counted is a length-$(2(r-\ell)-1)$ sequence that sums up to h, and we never count a sequence twice, which proves our claim.

To bound the seed length d, observe that for every $i + j \le h \le k$, either $i \le \lfloor k/2 \rfloor$ or $j \le \lfloor k/2 \rfloor$. If $i \le \lfloor k/2 \rfloor$, then $\mathbf{M}^{(i)}_{\ell..m}$ has a realization with seed length $(iA + B)$ by definition, and $\mathbf{M}^{(j)}_{m..r}$ has a realization with seed length $(jA + \log(r-m)B)$ by induction hypothesis. By Lemma 7.13, the seed length of the realization of $\mathbf{M}^{(i)}_{\ell..m}\mathbf{M}^{(j)}_{m..r}$ is at most

$$(i+j)A + (\log(r-m)+1)B \le hA + \log(r-\ell)B.$$

The $j \leq \lfloor k/2 \rfloor$ case is similar. By Lemma 7.14 and Lemma 7.15 we can then conclude that $d \leq hA + \log(r-\ell)B$.

Finally, the fact that ρ depends only on $\rho^{(k)}_{\ell..r}$ but not on $G^{(k)}_{\ell..r}$ follows from the proofs of Lemma 7.13, Lemma 7.14 and Lemma 7.15. □

From the previous lemma we see that if the base cases $\mathbf{M}^{(0)}_{\ell..r}$ have realizations of seed length B, then the seed length of the realization of the final outcome $\mathbf{M}^{(k)}_{0..n}$ grow to at least $\log(n)B$. If $\mathbf{M}^{(0)}_{\ell..r}$ is realized with Nisan's PRG, then $B = O(\log(n)\log(nw))$ and $\log(n)B$ is already too large. To reduce the seed length of the base cases, we apply samplers to get a realization that has shorter seed length but only works with high probability. The analysis is as follows.

Lemma 7.19. *Let $\mathbf{M} : \{0,1\}^n \to \mathbf{R}^{w \times w}$ be the transition matrix mapping of a (n,w)-ROBP. Let $\mathrm{Samp} : \{0,1\}^r \times \{0,1\}^{d'} \to \{0,1\}^d$ be a (ε,δ)-sampler, and (ρ, G) be a (K,d)-realization of a matrix $\widetilde{\mathbf{M}}$ based on $\mathbf{M}$. For every $q \in \{0,1\}^r$, define the matrix $\widetilde{\mathbf{M}}_{(q)}$ to be*

$$\widetilde{\mathbf{M}}_{(q)} := \sum_{i \in [K]} \rho(i) \mathop{\mathbb{E}}_{s \in \mathbf{U}_{d'}} \left[\mathbf{M}(G(i, \mathrm{Samp}(q,s)))\right].$$

Then with probability at least $1 - w^2\delta$ over $q \sim \mathbf{U}_r$,

$$\left\| \widetilde{\mathbf{M}}_{(q)} - \widetilde{\mathbf{M}} \right\|_\infty \leq wK\varepsilon.$$

Proof. For any $(a,b) \in [w] \times [w]$, consider the function $f_{a,b} : \{0,1\}^d \to \mathbb{R}$ defined as

$$f_{a,b}(t) := \left(\sum_{i \in [K]} \rho(i) \mathbf{M}\left(G(i,t)\right) \right)[a,b].$$

By definition, $\widetilde{\mathbf{M}}[a,b] = \mathbb{E}_{t \sim \mathbf{U}_d}\left[f_{a,b}(t)\right]$ and $\widetilde{\mathbf{M}}_{(q)}[a,b] = \mathbb{E}_{s \sim \mathbf{U}_{d'}}\left[f_{a,b}(\mathrm{Samp}(q,s))\right]$. In addition, observe that

$$\max_s f(s) - \min_s f(s) \leq |\{i \in [K] : \rho(i) = 1\}| + |\{i \in [K] : \rho(i) = -1\}| \leq K.$$

By definition of an (ε, δ)-sampler (Definition 2.18), we have that with probability at least $1 - \delta$ over $q \sim \mathbf{U}_r$,

$$\left|\widetilde{\mathbf{M}}_{(q)}[a,b] - \widetilde{\mathbf{M}}[a,b]\right| = \left|\mathop{\mathbb{E}}_{s\sim\mathbf{U}_{d'}}[f_{a,b}(\mathrm{Samp}(q,s))] - \mathop{\mathbb{E}}_{t\sim\mathbf{U}_d}[f_{a,b}(t)]\right| \leq K\varepsilon.$$

By union bound, the above inequality holds for every $(a,b) \in [w] \times [w]$ with probability at least $1 - w^2\delta$. Therefore, with probability at least $1 - w^2\delta$,

$$\left\|\widetilde{\mathbf{M}}_{(q)} - \widetilde{\mathbf{M}}\right\|_\infty = \max_{a\in[w]} \sum_{b\in[w]} \left|\widetilde{\mathbf{M}}_{(q)}[a,b] - \widetilde{\mathbf{M}}[a,b]\right| \leq wK\varepsilon.$$

□

Next we plug in the lemma above to the base-case matrices with parameters as in Lemma 7.18 and get the following corollary.

Corollary 7.20. *Suppose for some parameters $h \in \mathbb{N} \cup \{0\}$ and $0 < \varepsilon_0 < 1/8n^2$ and $\delta > 0$. Suppose there exists a $(h, d, \binom{2n-1}{h+1}\varepsilon_0^{h+1})$-natural WPRG (ρ, G) for (n,w)-ROBPs, where $K := \binom{2n+h-2}{h}$. Consider a (n,w)-ROBP and its transition matrices $\mathbf{M}_{[n]}$. Then there exists a collection of matrices $\{\widetilde{\mathbf{M}}_{(q)}\}_{q\in\{0,1\}^r}$ that satisfies the following.*

- $r = d + O(h\log(1/\varepsilon_0) + \log(w/\varepsilon_0\delta))$.
- *With probability at least $1 - \delta$ over $q \sim \mathbf{U}_r$, it holds that $\left\|\widetilde{\mathbf{M}}_{(q)} - \mathbf{M}_{0..n}\right\|_\infty \leq \binom{2n-1}{h+1}(1.5\varepsilon_0)^{h+1}$.*
- *Each $\widetilde{\mathbf{M}}_{(q)}$ has a (K, d')-realization (ρ, G_q), where $d' = O(h\log(1/\varepsilon_0) + \log(w/\varepsilon_0) + \log\log(1/\delta))$.*

Proof. If $n \leq 2h+1$, then we simply take $r = 0$ and $\mathbf{M}_{(0)} := \mathbf{M}_{0..n}$, which has a $(1,n)$-realization. One can verify that all the conditions are satisfied. Otherwise, let (ρ, G) be the natural WPRG as claimed, and define

$$\mathbf{M}^{(h)}_{0..n} := \sum_{i\in[K]} \rho(i) \mathop{\mathbb{E}}_{s\sim\mathbf{U}_d}[\rho(i)\mathbf{M}_{0..n}(G(i,s))]\,.$$

By definition of natural WPRG, $\left\|\mathbf{M}_{0..n} - \mathbf{M}^{(h)}_{0..n}\right\|_\infty \leq \binom{2n-1}{h+1}\varepsilon_0^{h+1}$. Then consider a $(\varepsilon_0^{h+1}/w, \delta/w^2)$-sampler $\mathrm{Samp} : \{0,1\}^r \times \{0,1\}^{d'} \to \{0,1\}^d$ from Lemma 2.25. By Lemma 2.25, it suffices to take $r = d' + O(h\log(1/\varepsilon_0) + \log(nw/\varepsilon_0\delta))$ and $d = O(h\log(1/\varepsilon_0) + \log(w/\varepsilon_0) + \log\log(n/\delta))$. Then define

$$\widetilde{\mathbf{M}}_{(q)} := \sum_{i\in[K]} \rho(i) \mathop{\mathbb{E}}_{s\in\mathbf{U}_{d'}} [\mathbf{M}(G(i, \mathrm{Samp}(q,s)))] .$$

Clearly $\widetilde{\mathbf{M}}_{(q)}$ has a (K, d')-realization (ρ, G_q) where $G_q(i,s) := G(i, \mathrm{Samp}(q,s))$, and such a realization is computable in space $S + O(\log\log(K) + r)$ (given q as a part of the input) by the fact that Samp is computable in space $O(r + d') = O(r)$. By Lemma 7.19, we can conclude that with probability at least $1-\delta$, it holds that

$$\begin{aligned}
\left\|\widetilde{\mathbf{M}}_{(q)} - \mathbf{M}_{0..n}\right\|_\infty &\leq \left\|\widetilde{\mathbf{M}}_{(q)} - \mathbf{M}^{(h)}_{0..n}\right\|_\infty + \left\|\mathbf{M}^{(h)}_{0..n} - \mathbf{M}_{0..n}\right\|_\infty \\
&\leq \binom{2n+h-2}{h}\varepsilon_0^{h+1} + \binom{2n-1}{h+1}\varepsilon_0^{h+1} \\
&= \binom{2n+h-2}{h+1}\varepsilon_0^{h+1} \cdot \left(\frac{h+1}{2n-2}\right) + \binom{2n-1}{h+1}\varepsilon_0^{h+1} \\
&\leq \binom{2n-1}{h+1}\varepsilon_0^{h+1} \cdot \left(\left(\frac{2n-1}{2n-h}\right)^h \cdot \left(\frac{h+1}{2n-2}\right) + 1\right) \\
&\leq \binom{2n-1}{h+1}(1.5\varepsilon_0)^{h+1} \qquad \text{(by } n \geq 2h+2\text{)}
\end{aligned}$$

□

Now we are ready to prove the following lemma, which can be applied repeatedly to get a reduction from WPRGs with small error to PRGs with moderate error.

Lemma 7.21. *Fix parameters $n, w, k \in \mathbb{N}$ and $0 < \varepsilon_0 < 1/8n^2$, where n is a power of 2. Suppose for every $n' < n$ and $k' \leq \lfloor k/2 \rfloor$ where n' is a power of 2, there is a $\left(\binom{2n'+k'-2}{k'}, d', \binom{2n'-1}{k'+1}\varepsilon_0^{k'+1}\right)$-natural WPRG for (n', w)-ROBPs. Then there is a (K, d, ε)-natural WPRG for (n, w)-ROBPs where $K := \binom{2n+k-2}{k}$, $\varepsilon := \binom{2n-1}{k+1}(2\varepsilon_0)^{k+1}$ and $d = d' + O(k\log(n/\varepsilon_0) + \log(n)\log(w/\varepsilon_0))$.*

Proof. If $n \le 10k + 10$ then we simply take the trivial $(1, n, 0)$-natural WPRG (ρ, G) defined as $\rho(1) = 1, G(x) = x$. Otherwise, fix a (n, w)-ROBP and its corresponding transition matrices $\mathbf{M}_{[n]}$. For every integers ℓ, r, k such that $(\ell, r) \in \mathsf{BS}_n \backslash \{(0, n)\}$ and $k' \le \lfloor k/2 \rfloor$, apply Corollary 7.20 on the subprorgram from layer ℓ to r with parameter $\delta = \varepsilon/8nKk$ and $h = k'$, and let $\{\mathbf{M}^{(k')}_{\ell..r,(q)}\}_{q \in \{0,1\}^{d_{\mathrm{out}}}}$ denote the corresponding collection.[24] Note that it suffices to take

$$d_{\mathrm{out}} = d' + O(k \log(1/\varepsilon_0) + \log(w/\varepsilon_0 \delta)) = d' + O(k \log(n/\varepsilon_0) + \log(w)).$$

In addition, each $\mathbf{M}^{(k')}_{\ell..r,(q)}$ has a $(\binom{2(r-\ell)+k'-2}{k'}, d^{(k')}_{\ell..r})$-realization, where

$$d^{(k')}_{\ell..r} = O(k' \log(1/\varepsilon_0) + \log(w/\varepsilon_0) + \log\log(1/\delta))) = k' \cdot O(\log(1/\varepsilon_0)) + O(\log(w/\varepsilon_0)).$$

By union bound, with probability at least $1 - \varepsilon/4K$ over $q \sim \mathbf{U}_{d_{\mathrm{out}}}$,[25] we have that $\left\|\mathbf{M}^{(k')}_{\ell..r,(q)} - \mathbf{M}_{\ell..r}\right\|_\infty \le \binom{2n-1}{k'+1}(1.5\varepsilon_0)^{k'+1}$ for every $(\ell, r) \in \mathsf{BS}_n \backslash \{(0, n)\}$ and $k' \le \lfloor k/2 \rfloor$. For every q, define $\mathbf{M}^{(k)}_{0..n,(q)}$ using the recursive formula in Lemma 7.18. Then by the second bullet point of Lemma 7.18, each $\mathbf{M}^{(k)}_{0..n,(q)}$ has a (K, d_{in})-realization (ρ, G_q) where $d_{\mathrm{in}} = O(k \log(1/\varepsilon_0) + \log(n) \log(w/\varepsilon_0))$. (Note that ρ does not depend on q.) Furthermore, with probability at least $1 - \varepsilon/4K$ over $q \sim \mathbf{U}_{d_{\mathrm{out}}}$, we have that $\left\|\mathbf{M}^{(k)}_{0..n,(q)} - \mathbf{M}_{0..n}\right\|_\infty \le \binom{2n-1}{k+1}(1.5\varepsilon_0)^{k+1} \le 3\varepsilon/4$ by the first bullet point of Lemma 7.18. Then define

$$\mathbf{M}^{(k)}_{0..n} := \sum_{i \in [K]} \rho(i) \mathop{\mathbb{E}}_{\substack{s \sim \mathbf{U}_{d_{\mathrm{in}}} \\ q \sim \mathbf{U}_{d_{\mathrm{out}}}}} \left[\mathbf{M}_{0..n}(G_q(s))\right].$$

Clearly $\mathbf{M}^{(k)}_{0..n}$ has a $(K, d_{\mathrm{in}} + d_{\mathrm{out}})$-realization (ρ, G) where $G(q \circ s) = G_q(s)$. One can verify that

$$d := d_{\mathrm{in}} + d_{\mathrm{out}} = d' + O(k \log(n/\varepsilon_0) + \log(n) \log(w/\varepsilon_0)).$$

[24] Here we assume all the collections have the same size by adding duplicates if necessary.
[25] Note that $|\mathsf{BS}_n| = 2n - 1$ and the number of choices of k' is at most $\lfloor k/2 \rfloor + 1 \le k$.

Furthermore, observe that $\mathbf{M}^{(k)}_{0..n} = \mathbb{E}_{q\sim\mathbf{U}_{d_{\text{out}}}}\left[\mathbf{M}^{(k)}_{0..n,(q)}\right]$, which implies

$$\begin{aligned}\left\|\mathbf{M}^{(k)}_{0..n} - \mathbf{M}_{0..n}\right\|_\infty &\le \mathop{\mathbb{E}}_{q\sim\mathbf{U}_{d_{\text{out}}}}\left[\left\|\mathbf{M}^{(k)}_{0..n,(q)} - \mathbf{M}_{0..n}\right\|_\infty\right] \\ &\le 3\varepsilon/4 + \frac{\varepsilon}{4K}\max_{q\in\{0,1\}^{d_{\text{out}}}}\left\|\mathbf{M}_{0..n,(q)} - \mathbf{M}_{0..n}\right\|_\infty \\ &\le \varepsilon.\end{aligned}$$

The last inequality follows from the fact that $\mathbf{M}_{0..n,(q)}$ is K-bounded, which implies $\left\|\mathbf{M}_{0..n,(q)} - \mathbf{M}_{0..n}\right\|_\infty \le K + 1 \le 2K$. Because the above analysis works for every (n, w)-ROBP (and its corresponding transition matrices $\mathbf{M}_{[n]}$), we can conclude that (ρ, G) is a (K, d, ε)-natural WPRG. □

Finally we apply the lemma repeatedly above to prove Theorem 7.7.

Proof of Theorem 7.7 (without space complexity analysis). Let $\varepsilon_0 = 1/n^3$. Suppose there exists an explicit ε_0-PRG for (n, w)-ROBPs with seed length d_0. This implies that for every $n' \le n$ there is a $(1, d_0, \varepsilon_0)$-natural WPRG for (n', w)-ROBPs. Let C be the constant factor in the big-O notation in Lemma 7.21. For every $k \in \mathbb{N}\cup\{0\}$, let $\operatorname{len}(k) := \lceil\log(k+1)\rceil$.[26] A simple case analysis shows that $\operatorname{len}(k) = \operatorname{len}(\lfloor k/2\rfloor) + 1$. We will prove by induction that for every $n' \le n$ which is a power of 2 and every $k \in \mathbb{N}\cup\{0\}$, there is a $(\binom{2n'+k-2}{k}, d_k, \binom{2n'-1}{k+1}(\varepsilon_k)^3)$-natural WPRG for (n', w)-ROBPs, where

$$d_k = d_0 + C\left(2k(\log(n/\varepsilon_0) + \operatorname{len}(k)) + \operatorname{len}(k)\log(n)\left(\log(w/\varepsilon_0) + \operatorname{len}(k)\right)\right)$$

and $\varepsilon_k = 2^{\operatorname{len}(k)}\varepsilon_0$. One can verify that the $k = 0$ base case is correct. The inductive case exactly follows from Lemma 7.21. The bound on boundedness and error is straightforward, and it remains to verify that

$$d_{\lfloor k/2\rfloor} + C(k\log(n/\varepsilon_{\lfloor k/2\rfloor}) + \log(n)\log(w/\varepsilon_{\lfloor k/2\rfloor})) \le d_k.$$

[26] We choose the notation len because $\operatorname{len}(k)$ is exactly the bit length of the binary representation, except that $\operatorname{len}(0) = 0$.

First note that for every $k \in \mathbb{N}$, $\mathrm{len}(k) = \mathrm{len}(\lfloor k/2 \rfloor) + 1$. Observe that

$$\begin{aligned} & C(k \log(n/\varepsilon_{\lfloor k/2 \rfloor}) + \log(n) \log(w/\varepsilon_{\lfloor k/2 \rfloor})) \\ & = C(k(\log(n/\varepsilon_0) + \mathrm{len}(k)) + \log(n)(\log(w/\varepsilon_0)) + \mathrm{len}(k)). \end{aligned}$$

and

$$d_{\lfloor k/2 \rfloor} \leq d_0 + C\left(k(\log(n/\varepsilon_0) + \mathrm{len}(k)) + (\mathrm{len}(k) - 1)\log(n)(\log(w/\varepsilon_0) + \mathrm{len}(k))\right).$$

Summing up the two inequalities above proves the claim. Now we obtain a $(\binom{2n+k-2}{k}, d_k, \binom{2n-1}{k+1}\varepsilon_k)$-natural WPRG for every $k \in \mathbb{N}$. Take a large enough k that implies $\binom{2n-1}{k+1}\varepsilon_k \leq (4k/n^2)^{k+1} \leq \varepsilon$. Note that if $\log(1/\varepsilon) > n/4$ we can take a trivial PRG with seed length $n = O(\log(1/\varepsilon))$. Otherwise it suffices to take $k = \log(1/\varepsilon)/\log(n) \leq n/4$, which also implies $\mathrm{len}(k) \leq \log(2k) \leq \log(n)$. By Lemma 7.12, the natural WPRG implies a WPRG that has seed length

$$\begin{aligned} d_k &= d_0 + O(k \log(kn/\varepsilon_0) + \log(k)\log(n)\log(nwk)) \\ &= d_0 + O(\log(1/\varepsilon) + \log\log_n(1/\varepsilon)\log(n)\log(nw)) \end{aligned}$$

and is $\binom{2n+k-2}{k} \leq O(n)^k = \mathrm{poly}(1/\varepsilon)$-bounded. □

7.5 Space Complexity Analysis

In this section we show that the WPRG construction discussed in the previous section is indeed explicit. Recall that the WPRG in Theorem 7.7 is constructed by repeatedly applying Lemma 7.21 to construct a $(K := \binom{2n+k-2}{k}, d, \varepsilon)$-natural WPRG (ρ, G) for some $k = O(\log(1/\varepsilon)/\log(n))$. First we prove the following lemma that describes some features of the recursive algorithm that computes (ρ, G).

Lemma 7.22. *Let P denote the 6-tuple $([n] \times ([k] \cup \{0\}))^2 \times [K] \times \{0,1\}^*$, and consider the following recursive algorithm that is defined based on some functions $F_L, F_R, F_S : P \to P$ and $V : P \to \{-1, 0, 1\}$.*

$\mathcal{A}(x := (n', k', n^*, k^*, e^*, s^*))$:

1. *If $n^* = 1$, append the first bit of s to the end of the output tape. Then return 1 if $e^* = 1$, or return 0 otherwise.*
2. *If $k' = 0$, append $G_{n',0}(s^*)$ to the end of the output tape, where $G_{n',0}$ is the base-case PRG for length n'. Then return 1 if $e^* = 1$, or return 0 otherwise.*
3. *If $k^* \leq \lfloor k'/2 \rfloor$, then recursively call and return $\mathcal{A}(F_S(x))$.*
4. *Otherwise, first check if $V(x) = 0$. If so, then append 0^{n^*} to the end of the output tape and return 0. Otherwise, recursively call and compute $v_L = \mathcal{A}(F_L(x))$ and $v_R = \mathcal{A}(F_R(x))$ in order, then return $v_L \cdot v_R \cdot V(x)$.*

Then there exist F_L, F_R, F_S, V such that for $e \in [K]$ and $s \in \{0,1\}^d$, $\mathcal{A}(n, k, n, k, e, s)$ print $G(e, s)$ on the output tape and return $\rho(e)$. In addition, the recursion has depth at most $O(\log(nk))$, and the functions F_L, F_R, F_S, V can be computed in space $O(d + \log(K))$.

Proof. We will prove by induction on (n', k') that $\mathcal{A}(n', k', n', k', e, s)$ correctly computes the natural WPRG from Lemma 7.21 with parameters n', k' (denoted by $(\rho_{n',k'}, G_{n',k'})$). We call this the "outer induction". The $n' = 1$ and $k' = 0$ base case of the outer induction is trivially correct. For the inductive case, let q be the prefix of s of length d_{out}. Recall that in the proof of *Lemma* 7.21, the function (ρ, G') where $G'(s') = G(q \circ s')$ is a realization of a matrix $\mathbf{M}^{(k')}_{0..n',(q)}$ that is defined using the recursion in Lemma 7.18. Fix any $(\ell, r) \in \mathsf{BS}_n$ such that $r - \ell = n^*$, and let (ρ^*, G^*) denote the (K^*, d^*)-realization of $\mathbf{M}^{(k^*)}_{\ell..r,(q)}$. We will prove by induction on (n^*, k^*) that $\mathcal{A}(n', k', n^*, k^*, e^*, q \circ s^*)$ prints $G^*(e^*, s^*)$ and returns $\rho^*(e^*)$. We call this the "inner

induction". Then the $n^* = n', k^* = k'$ case will prove that the outer induction is also correct.

First of all, note that the $n^* = 1$ base case is trivially correct. For the $k^* \leq \lfloor k'/2 \rfloor$ base case, note that from the proof of Lemma 7.21 and Corollary 7.20, we have $\rho^*(e^*) = \rho_{n^*,k^*}(e^*)$ and $G^*(e^*, s^*) = G_{n^*,k^*}(e^*, \mathrm{Samp}_{k^*}(q, s^*))$, where $\mathrm{Samp}_{k^*} : \{0,1\}^{d_{\mathrm{out}}} \times \{0,1\}^{d^*} \to \{0,1\}^{d'}$ is the $(\varepsilon_0^{k^*+1}/w, \delta/w^2)$-sampler that appears in the proof of Corollary 7.20. Therefore by defining

$$F_S(n', k', n^*, k^*, e^*, q \circ s^*) := (n^*, k^*, n^*, k^*, e^*, \mathrm{Samp}_{k^*}(q, s^*)),$$

the $k^* \leq \lfloor k'/2 \rfloor$ base case is also correct by hypothesis of the outer induction. In this case we simply choose the sequence to be $(n^*, k^*, e^*, \mathrm{Samp}_{k^*}(q, s^*))$ and set $\mathsf{sgn}^* = 1$. Otherwise, (ρ^*, G^*) is defined by the recursive formula

$$\mathbf{M}^{(k^*)}_{\ell..r,(q)} := \sum_{i+j=k^*} \mathbf{M}^{(i)}_{\ell..m,(q)} \cdot \mathbf{M}^{(j)}_{m..r,(q)} - \sum_{i+j=k^*-1} \mathbf{M}^{(i)}_{\ell..m,(q)} \cdot \mathbf{M}^{(j)}_{m..r,(q)}$$

and the matrix-to-realization mappings in Lemma 7.13, Lemma 7.14 and Lemma 7.15. This gives a way to recursively compute $\rho^*(e^*)$ and $G^*(s^*)$ as follows. To simplify notation, for every $0 \leq i \leq k^*$, let K_i denote $\binom{n^*+i-2}{i}$.

1. If $e^* \leq \sum_{i+j=k^*} K_i K_j$, let k_L be the minimum integer that satisfies $e^* \leq \sum_{i=0}^{k_L} K_i K_j$ and let $k_R = k^* - k_L$. Then define $e_{\mathrm{term}} := e - \sum_{i=0}^{k_L-1} K_i K_{k^*-i}$, $e_L := \lceil e_{\mathrm{term}}/K_{k_R} \rceil$ and $e_R = e^* - K_{k_R} \cdot (e_L - 1)$. Let (ρ_L, G_L) denote the (K_{k_L}, d_L)-realizations of $\mathbf{M}^{(k_L)}_{\ell..m,(q)}$ and (ρ_R, G_R) denote the (K_{k_R}, d_R)-realization of $\mathbf{M}^{(k_R)}_{m..r,(q)}$. In addition, let s_L be the length d_L-prefix of s^* and s_R denote the remaining suffix. Then we have $\rho^*(e^*) = \rho_L(e_L)\rho_R(e_R)$ and $G^*(e^*, s^*) = G_L(e_L, s_L) \circ G_R(e_R, s_R)$.
2. If $e^* > \sum_{i+j=k^*} K_i K_j$ and $e^* > \sum_{i+j=k^*} K_i K_j + \sum_{i+j=k^*-1} K_i K_j$, let k_L be the minimum integer that satisfies $e^* \leq \sum_{i+j=k^*} K_i K_j + \sum_{i=0}^{k_L} K_i K_{k^*-1-i}$ and let

$k_R = k^* - 1 - k_L$. Then define $e_{\text{term}} := e - \sum_{i+j=k^*} K_i K_j - \sum_{i=0}^{k_L-1} K_i K_{h-1-i}$, $e_L := \lceil e_{\text{term}}/K_{k_R} \rceil$ and $e_R = e^* - K_{k_R} \cdot (e_L - 1)$. Let (ρ_L, G_L) denote the (K_{k_L}, d_L)-realizations of $\mathbf{M}^{(k_L)}_{\ell..m,(q)}$ and (ρ_R, G_R) denote the (K_{k_R}, d_R)-realization of $\mathbf{M}^{(k_R)}_{m..r,(q)}$. In addition, let s_L be the length d_L-prefix of s^* and s_R denote the remaining suffix. Then we have $\rho^*(e^*) = \rho_L(e_L)\rho_R(e_R)$ and $G^*(e^*, s^*) = G_L(e_L, s_L) \circ G_R(e_R, s_R)$.

3. If $e' > \sum_{i+j=k^*} K_i K_j + \sum_{i+j=k^*-1} K_i K_j$, then $\rho^*(e^*) = 0$ and G^* is arbitrary.

Now define $V(x := (n', k', n^*, k^*, e^*, q \circ s^*))$ to be 1 in the first case, -1 in the second case, and 0 in the third case. Clearly $\mathcal{A}(x)$ is correct in the third case ($V(x) = 0$). In the first two cases, define

$$F_L(n', k', n^*, k^*, e^*, q \circ s^*) = (n', k', n^*/2, k_L, e_L, q \circ s_L)$$

and

$$F_R(n', k', n^*, k^*, e^*, q \circ s^*) = (n', k', n^*/2, k_R, e_R, q \circ s_R).$$

Then because $G^*(e^*, s^*) = G_L(e_L, s_L) \circ G_R(e_R, s_R)$ and $\rho^*(e^*) = V(x)\rho_L(e_L)\rho_R(e_R)$, the correctness of $\mathcal{A}(x)$ follows by the inductive hypothesis of the inner induction.

To bound the space complexity of F_L, F_R, F_S, v, observe that in the computation of $k_L, k_R, e_L, e_R, V(x)$ are all simple arithmetic operations that can be done in $O(\log K)$ space. To compute s_L, s_R, we may use the fact that either $k_L \leq \lfloor k'/2 \rfloor$ or $k_R \leq \lfloor k'/2 \rfloor$ and pick d_L (or d_R) to be the second input length of the sampler Samp_{k_L} (or Samp_{k_R}), which can be computed in $O(\log(d))$ space. In F_S we need to compute $\text{Samp}_{k^*}(q, s^*)$ which takes at most $O(d)$ space by Lemma 2.25.

Finally, note that the recursion depth is at most $O(\log(nk))$ because F_L, F_R, F_S decrease either k' or n^* by a factor of 2. □

It remains to show that the recursive algorithm $\mathcal{A}$ in Lemma 7.21 can be computed in space $O(d + \log(K))$. Note that we cannot simply store the parameter $x := (n', k', n^*, k^*, e^*, s^*)$ in all $O(\log(nk))$ levels because each x takes $O(d + \log(K))$ space. Nevertheless, we can instead store the "path" to the current recursion and compute x only when we need it.

Claim 7.23. *$\mathcal{A}$ in Lemma 7.21 can be computed in space $O(d + \log(K))$.*

Proof. We simulate $\mathcal{A}$ using a new recursive algorithm $\mathcal{A}'$ with the only difference being that we replace the local parameter x with a sequence $\mathsf{sq} \in \{L, R, S\}^*$ that is empty at the beginning. In addition, we use $O(d + \log(K))$ space to store global parameters (n, k, n, k, e, s). Whenever we make a recursive call to $\mathcal{A}$ with input $F_L(x)/F_R(x)/F_S(x)$, we replace it with a recursive call to $\mathcal{A}'$ with input being sq appended with $L/R/S$ respectively. Furthermore, whenever x is needed in $\mathcal{A}$, we compute it in $\mathcal{A}'$ with the following steps:

1. Initialize $x^* := (n, k, n, k, e, s)$.
2. For i from 1 to t where t is the length of sq, if the i-th letter in sq is $L/R/S$, then we update x^* to be $F_L(x^*)/F_R(x^*)/F_S(s^*)$.

Note that the computation above only takes $O(d + \log(K))$ space by Lemma 7.21. The other computation such as V and $G_{n',0}$ also takes at most $O(d + \log(K))$ space. In addition, to store sq in all $O(\log(nk)) = O(\log(n))$ levels, it takes only $O(\log^2(n)) = O(d)$ space.[27] Therefore the total space complexity is $O(d + \log(K))$. □

[27] We can in fact store sq as a global stack, and push a new letter $L/R/S$ to it whenever we make a new recursive call and pop it after the recursive call. This takes only $O(\log(n))$ space instead of $O(\log^2(n))$.

7.6 Saks-Zhou Scheme on WPRGs

In this section, we first briefly sketch Saks and Zhou's proof for $\mathbf{BPL} \subseteq \mathbf{L}^{3/2}$ [SZ99] and Armoni's trick for replacing Nisan's PRG with any PRG in the Saks-Zhou scheme [Arm98]. (We recommend interested readers to check [SZ99, Arm98] for formal proofs and also [HU17] for a beautiful summary.) Then we will see why a $\mathrm{poly}(w/\varepsilon)$-bounded WPRG suffices.

7.6.1 Saks and Zhou's Scheme

It is well-known that derandomizing $\mathbf{BPL}$ can be reduced to approximating the the n-th power of a $n \times n$ stochastic matrix. The first step of Saks and Zhou is to turn M^n into the following recursive computation:

Fact 7.24. *Let n_1, n_2 be integers such that $n_1^{n_2} = n$. Define $M_0 = M$, and $M_i = M_{i-1}^{n_1}$ for every positive integer i. Then $M_{n_2} = M^n$.*

To approximate M_{n_2}, it suffices to compute $M_i = M_{i-1}^{n_1}$ with small enough error in each step. However, if we need s bits of space to approximate the n_1-th power of a stochastic matrix, we will need $O(sn_2)$ bits of space in total. This doesn't really save any space (over approximating M^n directly) if we approximate $M_{i-1}^{n_1}$ with PRGs such as Nisan's generators. The first idea of Saks and Zhou is to utilize the "high probability property" of Nisan's generator:

Lemma 7.25 ([Nis92])**.** *For every n, w, ε there exists an algorithm $\widehat{\mathrm{Pow}}_n$ which takes a $w \times w$ (sub)stochastic matrix M and a string $y \in \{0,1\}^{O(\log n \log(nw/\varepsilon))}$ as input, and outputs a $w \times w$ matrix such that*

$$\Pr_y \left[\left\| \widehat{\mathrm{Pow}}_n(M, y) - M^n \right\|_{\max} < \varepsilon \right] \geq 1 - \varepsilon$$

in space $O(log(nw/\varepsilon))$.

In other words, derandomization with Nisan's generator has the following structure. First it fixes an "offline randomness" $y \in \{0,1\}^r$ and considers it as a part of input. With a fixed randomness y, it takes only s bits of additional "processing space" to compute an approximation of M^n, where $s \ll r$, and the output is guaranteed to be a good approximation with high probability over y. (This is called an "offline randomized algorithm" in [SZ99].) With these properties, the main idea of Saks and Zhou is to reuse the same offline randomness for each level of recursion. If computing M^{n_1} takes r bits of offline randomness and s bits of processing space, then computing M^n will take r bits of offline randomness and $O(sn_2)$ bits of processing space. The space complexity would be $O(r + sn_2)$, which is better than approximating M^n directly since the offline randomness part was the original bottleneck.

However, there's a problem in this construction: if we compute $\widehat{M_1} = \widehat{\mathrm{Pow}}_{n_1}(M, y)$, and try to use $\widehat{\mathrm{Pow}}_{n_1}(\widehat{M_1}, y)$ to approximate $M_2 = M_1^{n_1}$, it might be possible that $\widehat{\mathrm{Pow}}_{n_1}(\widehat{M_1}, y)$ is always a bad approximation because $\widehat{M_1}$ depends on y. To deal with this issue, the second idea of Saks and Zhou is to break the dependency with a randomized shift-and-truncate operation. Here we borrow the name "snap" from [HU17] for this operation.

Definition 7.26. *Given value* $x \in \mathbb{R}$, *string* $y \in \{0,1\}^d$, *define*

$$\mathrm{Snap}_d(x, y) = \max(\lfloor x \cdot 2^d - 2^{-d} y \rfloor \cdot 2^{-d}, 0).$$

For a $w \times w$ *matrix* M, *define* $\mathrm{Snap}_d(M, y)$ *to be the matrix* M' *such that* $M'_{i,j} = \mathrm{Snap}_d(M_{i,j}, y)$ *for every* $i, j \in [w]$.

It's not hard to prove the following lemma:

Lemma 7.27 ([SZ99]). *For any matrix M, M' such that $\|M - M'\|_{\max} \le \varepsilon$,*

$$\Pr_y [\mathrm{Snap}_d(M, y) \neq \mathrm{Snap}_d(M', y)] \le w^2(2^d\varepsilon + 2^{-d}).$$

Proof. The snap operation is equivalent to randomly choose a grid of length 2^{-d} and round each value to the closest grid point the left. Therefore two values a, b are rounded to different points only if there is a grid point between them, which happens with probability at most $2^d|a - b| + 2^{-d}$. By union bound and the fact that $\|M - M'\|_{\max} \le \varepsilon$ the lemma follows. □

With the lemma above, we can see that by taking $\widehat{M_1} = \mathrm{Snap}_d(\widehat{\mathrm{Pow}}_{n_1}(M, y), z)$ instead, $\widehat{M_1}$ will be equivalent to $\mathrm{Snap}_d(M^{n_1}, z)$ with high probability, which is independent of y. Therefore we can use y as the offline randomness to compute the n_1-th power of $\widehat{M_1}$. Moreover, if the rounding precision (d) is high enough, the snapped matrix is still a good approximation. Finally we get Saks-Zhou theorem.

Lemma 7.28 ([SZ99]). *Let n_1, n_2 be integers such that $n_1^{n_2} = n$. Suppose there exists an offline randomized algorithm $\widehat{\mathrm{Pow}}_{n_1}$ which takes r bits of randomness and s bits of processing space such that for every substochastic matrix M,*

$$\Pr_x \left[\left\| \widehat{\mathrm{Pow}}_{n_1}(M, y) - M^{n_1} \right\|_{\max} \le \varepsilon \right] \ge 1 - \varepsilon.$$

Now consider uniform random bits $y \in \{0,1\}^r$ and $z_1, z_2, \ldots, z_{n_2} \in \{0,1\}^d$. Let $\widehat{M_0} = M$, and $\widehat{M_i} = \mathrm{Snap}_d(\widehat{\mathrm{Pow}}_{n_1}(\widehat{M_{i-1}}, y), z_i)$ for every $i \in [n_2]$. Then with probability at least $1 - O(w^2 n_2(2^d\varepsilon + 2^{-d}))$ over $y, z_1, \ldots, z_{n_2}$,

$$\left\| \widehat{M_{n_2}} - M^n \right\|_\infty \le nw2^{-d+1}.$$

Moreover, the space complexity of computing $\widehat{M_{n_2}}$ is $O(r + n_2(s + d))$.

Proof (sketch). Define $\overline{M_0} = M$, $\overline{M_i} = \mathrm{Snap}((\overline{M_{i-1}})^{n_1}, z_i)$. By union bound, the following events happen simultaneously with probability $1 - O(w^2 n_2(2^d\varepsilon + 2^{-d}))$:

1. For every $i \in [n_2]$, $\left\| \widehat{\mathrm{Pow}}_{n_1}(\overline{M_{i-1}}, y) - (\overline{M_{i-1}})^{n_1} \right\|_{\max} \leq \varepsilon$.
2. For every $i \in [n_2]$, conditioned on $\widehat{M_{i-1}} = \overline{M_{i-1}}$, $\widehat{M_i} = \overline{M_i}$.

When the above events occur, we have $\widehat{M_{n_2}} = \overline{M_{n_2}}$. Moreover, note that for every $i \in [n_2]$

$$\left\| \overline{M_i} - (\overline{M_{i-1}})^{n_1} \right\|_{\max} \leq 2^{-d+1}.$$

To see why this is true, observe that in a snap operation we change the given value by at most 2^{-2d} from perturbation and 2^{-d} from rounding. [28] This implies

$$\left\| \overline{M_i} - (\overline{M_{i-1}})^{n_1} \right\|_{\infty} \leq 2^{-d+1}.$$

Note that a snap operation can only decrease the entries of a matrix, so every $\overline{M_i}$ has infinity norm at most 1. Therefore one can repeated apply Lemma 2.47 and sub-additivity to that

$$\left\| \overline{M_{n_2}} - M^n \right\|_{\infty} \leq \left(\sum_{i=0}^{n_2-1} n_1^i \right) 2^{-d+1} \leq n 2^{-d+1}.$$

For the space complexity, observe that we can compute $\widehat{M_{n_2}}$ with n_2 levels of recursive calls, and each recursive call takes $O(s+d)$ bits. Moreover, we need r bits to store the offline randomness. Therefore the space complexity is $O(n_2(s+d)+r)$ $\square$

If we take $n_2 = \sqrt{\log n}$, $n_1 = 2^{\sqrt{\log n}}$, $d = O(\log(n))$ and $\varepsilon = 2^{-2d}$ and plugging in Nisan's generator, the above lemma shows that $\mathbf{BPL} \subseteq \mathbf{L}^{3/2}$.

7.6.2 Armoni's Trick

We saw that in Saks and Zhou's proof, we need a "offline randomized algorithm" for substochastic matrix exponentiation such that when given r bits of randomness

[28] Note that capping the lowest possible value to be 0 can only reduce the error, because the snapped value was non-negative.

as additional input, the algorithm only requires additional $s \ll r$ bits of space to compute a good approximation with high probability. This is in fact the only place where we need PRGs in Saks and Zhou's proof. However, not every PRG has such property, so it might be hard to tell whether an improvement over Nisan's PRG will actually give a better derandomization for **BPL**. Fortunately, Armoni [Arm98] observed that one can turn any PRG into a derandomization algorithm with the required property by simply composing the PRG with an averaging sampler.

Before we go through Armoni's claim, first we generalize Lemma 7.28 for $\widehat{\mathrm{Pow}}(M, y)$ that has small space complexity whenever reading a bit from M. (See the discussions in Section 2.7.)

Definition 7.29. *Consider a offline randomized algorithm $F(x, y)$, where y is the offline randomness. We say F has* sensitive space complexity t *if whenever F reads from x, only the first t cells on the work tape is occupied.*

Lemma 7.30 ([SZ99], generalized)**.** *Let n_1, n_2 be integers such that $n_1^{n_2} = n$. Suppose there exists an algorithm $\widehat{\mathrm{Pow}}_{n_1}$ such that for every substochastic matrix M,*

$$\Pr_y \left[\left\| \widehat{\mathrm{Pow}}_{n_1}(M, y) - M^{n_1} \right\|_{\max} \le \varepsilon \right] \ge 1 - \varepsilon.$$

In addition, $\widehat{\mathrm{Pow}}_{n_1}$ has space complexity s and sensitive space complexity t. Now consider uniform random bits $y \in \{0,1\}^r$ and $z_1, z_2, \ldots, z_{n_2} \in \{0,1\}^d$. Let $\widehat{M_0} = M$, and $\widehat{M_i} = \mathrm{Snap}_d(\widehat{\mathrm{Pow}}_{n_1}(\widehat{M_{i-1}}, y), z_i)$ for every $i \in [n_2]$. Then with probability at least $1 - O(w^2 n_2 (2^d \varepsilon + 2^{-d}))$ over $y, z_1, \ldots, z_{n_2}$,

$$\left\| \widehat{M_{n_2}} - M^n \right\|_\infty \le nw2^{-d+1}.$$

Moreover, the space complexity of computing $\widehat{M_{n_2}}$ is $O(r + s + n_2(t + d))$.

Proof. Use the exact same proof of Lemma 7.28. The space complexity bound follows from Lemma 2.47. □

For technicality, we also need to define ROBPs that read multiple bits at a time.

Definition 7.31. *A (n, w, d)-ROBP is specified by a start state $v_0 \in [w]$, a set of accept states V_{acc} and n transition functions $B_i : [w] \times \{0,1\}^d \to [w]$ for i from 1 to n. The ROBP B computes a function $B : \{0,1\}^n \to \{0,1\}$ as follows. Given an input $x = (x_1, \ldots, x_d) \in (\{0,1\}^d)^n$, define $v_i = B_i(v_{i-1}, x_i)$ for every $i \in [n]$. Then output $B(x) = 1$ if $v_n \in V_{\mathrm{acc}}$, or $B(x) = 0$ otherwise.*

The "step size" d corresponds to the number of bits that is used to represent an entry in the stochastic matrices $\widehat{M_i}$.

Now we are ready to introduce Armoni's Lemma.

Lemma 7.32 ([Arm98])**.** *Suppose there exists an explicit PRG for $(n, w+1, \log(3nw/\varepsilon))$-ROBP with error $\varepsilon/3$ which has seed length s. Then there exists an offline randomized algorithm $\widehat{\mathrm{Pow}}(M, y)$ which approximates the n-th power of a substochastic matrix M within error ε (in max norm) with probability at least $1 - \varepsilon$ over the randomness $y \in \{0,1\}^{s+O(\log(w/\varepsilon))}$. Moreover, $\widehat{\mathrm{Pow}}$ has space complexity $O(s + O(\log(w/\varepsilon)))$ and sensitive space complexity $O(\log(nw/\varepsilon))$.*

Proof. Given an input M, first we round each entry down to $d = \log(3nw/\varepsilon)$ bits of precision. Then we will get a substochastic matrix M' such that each entry of M' is a multiple of 2^{-d}, and $\|M - M'\|_{\max} \le \varepsilon/3nw$. Then we have

$$\|M^n - (M')^n\|_{\max} \le \|M^n - (M')^n\|_\infty \le n \|M - M'\|_\infty \le nw \|M - M'\|_{\max} \le \frac{\varepsilon}{3}.$$

Then we construct a $(n, w+1, d)$-ROBP B as follows. For every $t \in [n], i \in [w], x \in \{0,1\}^d$, first interpret x as a number in $[2^d]$, and define $B_t(i, x) = \arg\min_k(\sum_{j \le k} M'_{i,j} \cdot 2^d \ge x)$. To make sure that $B_t(i, x)$ is well-defined we define $M'_{i,w+1} = 1 - \sum_{j \in [w]} M'_{i,j}$. (Note that each $M'_{i,j}$ is a multiple of 2^{-d}, so there are exactly $M'_{i,j} \cdot 2^d$ different x that

satisfy $B_t(i,x) = j$.) Then we simply define $B_t(w+1,x) = w+1$. It is easy to observe that $(M'^n)_{i,j}$ is exactly the probability that we start a random walk from $v_0 = i$ and reach $v_n = j$. Now for every $i,j \in [w]$, define $B_{i,j}(x)$ to be the indicator for whether we will reach $v_n = j$ if we start from $v_0 = i$ and follow $x \in (\{0,1\}^d)^n$. Then $\mathbb{E}_x[B_{i,j}(x)] = (M'^n)_{i,j}$. Take the given PRG G, we have

$$\left|\mathop{\mathbb{E}}_r[B_{i,j}(G(r))] - \mathop{\mathbb{E}}_x[B_{i,j}(x)]\right| \le \frac{\varepsilon}{3}.$$

Now define the offline randomized algorithm $\widehat{\mathrm{Pow}}$ to be

$$\widehat{\mathrm{Pow}}(M,y)_{i,j} = \mathop{\mathbb{E}}_z[B_{i,j}(G(\mathrm{Samp}(y,z)))],$$

where Samp is a $(\varepsilon/3, \varepsilon/w^2)$-sampler. By definition of sampler, with probability at least $(1-(\varepsilon/w^2))$ over the choice of y, we have

$$\left|\widehat{\mathrm{Pow}}(M,y)_{i,j} - \mathop{\mathbb{E}}_r[B_{i,j}(G(r))]\right| \le \frac{\varepsilon}{3}.$$

By union bound, with probability at least $(1-\varepsilon)$,

$$\left\|\widehat{\mathrm{Pow}}(M,y)_{i,j} - \mathop{\mathbb{E}}_r[B_{i,j}(G(r))]\right\|_{\max} \le \frac{\varepsilon}{3}$$

for every $i,j \in [w]$. Therefore by sub-additivity we have

$$\left\|\widehat{\mathrm{Pow}}(M,y) - M^n\right\|_{\max} \le \varepsilon$$

with probability at least $1-\varepsilon$.

Finally we compute the complexity of $\widehat{\mathrm{Pow}}$. By Lemma 2.25, the required number of random bits is $s + O(\log(1/\varepsilon) + \log\log(w/\varepsilon))$. The space complexity is sum of space complexity for samplers and PRGs. To bound the sensitive space complexity, observe that when we read from M, we only need to store the second input of the sampler (z); the current node in the ROBP, which takes $\log(nw)$ bits to store; and a d-bit block in $G(\mathrm{Samp}(y,z))$ which is the input that we read in the current transition

functions. Therefore the required sensitive processing space is only $O(\log(nw/\varepsilon))$ bits. □

With Armoni's sampler trick, if we have any PRG for $(n, w+1, \log(3nw/\varepsilon))$-ROBP, we can always plug it into the Saks-Zhou scheme regardless of whether it has the high-probability property. Specifically, as suggested in [BCG20], if we have a PRG of seed length $O(\log^2(n) + \log^{4/3}(w/\varepsilon))$, we can even prove that $\mathbf{BPL} \subseteq \mathbf{L}^{4/3}$.

7.6.3 Saks-Zhou-Armoni Scheme with WPRGs

Finally we see how to apply a WPRG in the above scheme.

Lemma 7.33. *Suppose there exists an explicit* $\mathrm{poly}(nw/\varepsilon)$*-bounded WPRG* (ρ, G) *for* $(n, w+1, \log(3nw/\varepsilon))$*-ROBP with error* $\varepsilon/3$ *which has seed length* s*. Then there exists an offline randomized algorithm which approximates the* n*-th power of any substochastic matrix within error* ε *with probability at least* $1-\varepsilon$*. Moreover, such algorithm requires* $s + O(\log(w/\varepsilon))$ *bits of offline randomness, and has space complexity* $O(s + O(\log(w/\varepsilon)))$ *and sensitive space complexity* $O(\log(nw/\varepsilon))$*.*

Proof. The proof is basically the same as Lemma 7.32, with the following two difference.

- $\widehat{\mathrm{Pow}}(M, y)_{i,j}$ is defined as $\mathbb{E}_z\left[\rho(\mathrm{Samp}(y, z)) \cdot B_{i,j}(G(\mathrm{Samp}(y, z)))\right]$ instead.
- If (G, ρ) is k-bounded, then we will choose Samp as a $(\varepsilon/6k, \varepsilon/w^2)$ sampler instead.[29]

[29] The reason we need to choose the error to be $\varepsilon/6k$ instead is because we want the sampler to fool the function $f(x) := \rho(x) \cdot B_{i,j}(G(x))$, which has range $[-k, k]$. To apply the standard definition of sampler that works for functions with range $[0, 1]$ we need to consider the function $f' = (f+k)/2k$ instead, and $\varepsilon/2$ error for f is $\varepsilon/6k$ error for f'.

The sensitive space complexity is increased to $O(\log(nw/\varepsilon) + \log(k))$, which is still $O(\log(nw/\varepsilon))$ if $k = \mathrm{poly}(nw/\varepsilon)$. □

One may notice that there might have negative output in our new definition of $\widehat{\mathrm{Pow}}$. However, this is not a problem when applying Saks-Zhou argument because we only rely on the non-negativeness of matrices $\overline{M_i}$, which is independent of the approximation algorithm we use. With the above lemma we have the following corollary, which better motivates the problem of getting improved seed length for WPRGs.

Corollary 7.34. *If there exists a* $\mathrm{poly}(nw/\varepsilon)$*-bounded explicit PRPD for* (n, w, d)*-ROBP with error* ε *which has seed length* $O(\log^2(n) + (\log(w/\varepsilon) + d)^{4/3})$*, then* $\mathbf{BPL} \subseteq \mathbf{L}^{4/3}$*.*

Proof. Apply the Saks-Zhou scheme (Lemma 7.30), and take $n_1 = 2^{\log^{2/3}(n)}$, $n_2 = \log^{1/3}(n)$, $d = 10\log(n)$ and $\varepsilon = 2^{-2d}$. The required subprocedure $\widehat{\mathrm{Pow}}$ would be approximating the n_1-th power of $n \times n$ substochastic matrices within error ε. By Lemma 7.33, there exists an offline randomized algorithm which approximates M^{n_1} within error $\varepsilon = 2^{-2d} = \mathrm{poly}(1/n)$, which requires sensitive processing space $O(\log(n))$ and offline randomness + reusable processing space $O(\log^2(n_1) + \log^{4/3} n) = O(\log^{4/3}(n))$. Therefore the total space complexity is $O(\log(n) \cdot n_2 + \log^{4/3}(n)) = O(\log^{4/3}(n))$. □

Remark 7.35. *Note that while we only construct WPRGs for* (n, w)*-ROBP, it is possible to adapt our construction to get a reduction from WPRGs to PRGs for* (n, w, d)*-ROBP. (The only difference is in the* $r - \ell = 1$ *base case.) Since it doesn't imply better derandomization for* $\mathbf{BPL}$ *anyway, we keep* $d = 1$ *for simplicity.*

Chapter 8

Error Reduction for Regular ROBPs

In the previous section we saw how to reduce WPRG with small error to PRG with larger error. In this section, we discuss some technical difficulties of applying this error reduction technique in the setting of regular branching programs. This problem was first resolved by Chen, Hoze, Lyu, Tal and Wu [CHL+23]. In this chapter we show that the recursive formula we introduced in the previous section gives a much simpler solution to this problem.

8.1 Introduction

As we discussed in the previous section, the best known PRG construction for ROBPs is still Nisan's three-decade-old result [Nis92]. While there has been no improvement over Nisan's seed length for general ROBPs, a lot of progress has been

The results presented in this chapter is based on the following joint work:

- [CL24] Eshan Chattopadhyay and Jyun-Jie Liao. Recursive error reduction for regular branching programs. In *15th Innovations in Theoretical Computer Science Conference, ITCS 2024*, volume 287 of *LIPIcs*, pages 29:1–29:20, 2024

made in some restricted families. One important example is the setting of *regular ROBPs*, which is the main focus of this chapter.

Definition 8.1 (Regular ROBPs)**.** *We say a (standard-order) ROBP B is* regular *if for every transition function $B_i : [w] \times \{0,1\} \to [w]$ in B, every state $v \in [w]$ has exactly 2 pre-images.*

An important reason to study this family is that general ROBPs can be reduced to regular ROBPs [RTV06, BHPP22]. In fact, a surprisingly simple proof in a recent work by Lee, Pyne and Vadhan [LPV23] shows that any function computable with a ROBP of length n and width w can also be computed by a regular ROBP of width $O(nw)$.

In 2010, Braverman, Rao, Raz and Yehudayoff [BRRY14] proved that the INW generator [INW94] (a variant of Nisan's PRG) with proper choices of parameters is in fact a PRG for regular ROBPs with seed length $O(\log(n) \cdot (\log\log(n) + \log(w/\varepsilon)))$. This is better than the seed length of Nisan's PRG when $\log(w/\varepsilon) = o(\log(n))$. Specifically, they introduced the "weight" measure for ROBPs and proved that an INW generator with fixed parameters has error proportional to the weight. Their better seed length bound then follows by showing that regular ROBPs have smaller weight than general ROBPs when $w \ll n$. (See Section 8.3 for the formal statements.) With the argument above, their result can be generalized to work for "small-weight ROBPs", which turns out to be an important ingredient of the PRG for width-3 ROBPs in [MRT19].

Recently, another remarkable result of derandomizing regular ROBPs was proved by Ahmadinejad, Kelner, Murtagh, Peebles, Sidford and Vadhan [AKM+20]: they showed that it takes only $\tilde{O}(\log(nw))$ space to estimate the expectation of a regular ROBP B in a non-black-box way. In fact, they designed an algorithm that can

estimate the expectation of B to a very high precision without much overhead:

Theorem 8.2. *For every $\varepsilon > 0$ there is a deterministic algorithm which takes a regular ROBP B of length n and width w as input, and computes a value within $\mathbb{E}_x[B(x)] \pm \varepsilon$ in space complexity $\tilde{O}(\log(nw)\log\log(1/\varepsilon))$.*

8.1.1 Error Reduction for Regular ROBPs

Given the better PRG by [BRRY14] in the regular setting, it is natural to ask whether one can get a better WPRG than [Hoz21a] in the regular setting as well. It seems quite plausible that the answer should be yes: we saw in Theorem 7.7 that there ε-WPRG can be reduced to ε_0-PRG for some $\varepsilon_0 \gg \varepsilon$ in a black-box way, and in the general setting we take the base PRG to be Nisan's PRG. Therefore, if we take the PRG in [BRRY14] as the base PRG instead, it seems very likely that we should also get a better WPRG in the regular setting.

However, it turns out that the intuition is not trivially true, because every known error reduction procedure for general ROBPs (including Theorem 7.7 and those in [CDR+21, PV21, Hoz21a]) requires the "base-case error" ε_0 to be $1/\operatorname{poly}(n)$. When $\varepsilon_0 < 1/n$, the $\tilde{O}(\log(n)\log(w/\varepsilon_0))$ seed length bound in [BRRY14] is no longer better than Nisan's $O(\log(n)\log(nw/\varepsilon_0))$ seed length. As a result, we do not get any improvement in the seed length of the corresponding WPRG.

This problem was recently solved by Chen, Hoza, Lyu, Tal and Wu [CHL+23]. They showed how to exploit the regular property and obtain a reduction from ε-WPRG for regular ROBPs to PRG for regular ROBPs with error $\varepsilon_0 = O(1/\log^2(n))$. As a result, they proved the following theorem. (For the rest of this chapter, we refer to their construction as the "CHLTW construction".)

Theorem 8.3 ([CHL+23])**.** *There is an explicit ε-WPRG for regular ROBPs with seed length*

$$\tilde{O}\left(\log(n)\left(\log(w)+\sqrt{\log(1/\varepsilon)}\right)+\log(1/\varepsilon)\right).$$

Following [AKM+20, CDR+21, PV21, Hoz21a], the WPRG construction in [CHL+23] is based on the "inverse Laplacian" perspective of small-space derandomization and Richardson iteration. The key step in their construction is to modify the approximated inverse Laplacian based on a structure called "shortcut graph". With the shortcut graph structure, they showed how to apply the potential argument in [BRRY14] to get a better bound for ε that is still non-trivial even when $\varepsilon_0 = O(1/\log^2(n))$. Based on the same idea, [CHL+23] also showed how to get a simplified proof of the non-black-box derandomization result in [AKM+20] (Theorem 8.2).

In short, the main purpose of using the shortcut graph idea in [CHL+23] is to embed a "binary-recursive-like" structure into the inverse Laplacian analysis. Such a structure makes their analysis compatible with the potential argument in [BRRY14]. In order to prove the non-black-box derandomization result in Theorem 8.2, [CHL+23] showed that one can apply a different potential argument based on the notion of "singular-value approximation" (SV approximation) defined by Ahmadinejad, Peebles, Pyne, Sidford and Vadhan [APP+23].

8.1.2 Our Contribution

While the shortcut graph modification gives a nice structure to the inverse Laplacian analysis, the inverse Laplacian perspective itself is sometimes tricky to work with. In fact, although the proof of Theorem 8.2 in [CHL+23] is simpler than the original

proof in [AKM+20], they still need to work on a sophisticated matrix seminorm, and the corresponding potential argument requires non-trivial ideas to analyze.

In this chapter, we give an alternative solution to this problem: we prove that for *regular* ROBPs, the same recursive construction in Lemma 7.17 can work for base-case error $\varepsilon_0 < O(1/\log(n))$ (instead of $1/\operatorname{poly}(n)$ as in Lemma 7.17).

Claim 8.4 (informal). *Consider a* regular (n, w)*-ROBP and its transition matrices* $\mathbf{M}_{[n]}$ *as defined in Section 7.2.*[30] *Fix a parameter* $\varepsilon_0 = O(1/\log(n))$*. For every* $(\ell, r) \in \mathsf{BS}_n$*, let* $\mathbf{M}^{(0)}_{\ell..r}$ *be a matrix such that the error of* $\mathbf{M}^{(0)}_{\ell..r}$ *from* $\mathbf{M}_{\ell..r}$ *is at most* ε_0*. For every* $k \in \mathbb{N}$ *and every* $(\ell, r) \in \mathsf{BS}_n$*, recursively define*

$$\mathbf{M}^{(k)}_{\ell..r} := \begin{cases} \mathbf{M}_r & \text{if } r - \ell = 1, \\ \sum_{i+j=k} \mathbf{M}^{(i)}_{\ell..m} \cdot \mathbf{M}^{(j)}_{m..r} - \sum_{i+j=k-1} \mathbf{M}^{(i)}_{\ell..m} \cdot \mathbf{M}^{(j)}_{m..r} & \text{otherwise}, \end{cases}$$

where $m = (\ell + r)/2$*. Then for every* $k \in \mathbb{N}$*, the error of* $\mathbf{M}^{(k)}_{0..n}$ *from* $\mathbf{M}_{0..n}$ *is at most* $O(\log(n)\varepsilon_0)^{k+1}$*.*

The formal definition of "error" depends on the applications, which we will discuss in the later sections. Note that the $O(\log(n)\varepsilon_0)^{k+1}$ error bound here is much better than the $O(n\varepsilon_0)^{k+1}$ bound for general ROBPs in Lemma 7.17.

The advantage of this alternative construction is that it is *actually binary recursive*, and thus is naturally compatible with the weight argument in [BRRY14]. To construct a WPRG for regular branching program that matches the parameter in Theorem 8.3, we show that the analysis in Lemma 7.17 can be improved in the regular setting based on the weight argument in [BRRY14]. Inspired by the proof of Theorem 8.2 in [CHL+23], we also give an alternative proof of Theorem 8.2 based on the notion of SV approximation [APP+23]. Because of the binary recursive nature

[30] Again we assume w.l.o.g. that n is a power of 2.

of our construction, both proofs are relatively straightforward by induction and are arguably simpler than the proofs in [CHL+23].

In fact, our proof of Theorem 8.2 implies a slightly stronger claim (Theorem 8.24) which might be of independent interest: we can compute an ε-SV approximation of the random walk matrix of any regular ROBP of width w and length n in space $\tilde{O}(\log(nw)\log\log(1/\varepsilon))$. (See [APP+23] for comparison between SV approximation and other notions of approximation.) It is not clear how to obtain this stronger claim from the previous proofs of Theorem 8.2 [AKM+20, CHL+23].

Finally, we show in Section 8.5 that the Laplacian-based construction in [CHL+23] is actually equivalent to our recursive construction. We find this fact surprising because the two constructions are defined in very different ways. (Our proofs of Theorem 8.2 and Theorem 8.3 are self-contained and do not rely on this fact.)

Remark 8.5. *There are two additional results in [CHL+23] which are based on their proof of Theorem 8.2 and Theorem 8.3: WPRGs for width-3 ROBPs and WPRGs for unbounded-width permutation ROBPs, both having seed length $\tilde{O}(\log(n)\sqrt{\log(1/\varepsilon)}+\log(1/\varepsilon))$. Our new proofs for Theorem 8.2 and Theorem 8.3 can also be plugged into the corresponding parts of their proofs to get the same results.*

8.2 Proof Overview

Similar to [CHL+23], the reason why we can get an improved error bound in the regular setting is because a regular ROBP has a bounded "total amount of mixing", no matter how large n is. With this property, we inductively prove an approximation guarantee that the error of $\mathbf{M}^{(k)}_{\ell..r}$ is *proportional to the amount of mixing* from layer ℓ

to layer r. For the proof of WPRG construction (Theorem 8.3), this statement is formalized based on the "weight" defined in [BRRY14]. For the proof of non-black-box derandomization (Theorem 8.2), this statement is formalized with the notion of SV approximation [APP+23]. We defer the formal definitions to later sections, and focus on why this statement gives a better bound.

For every $(i,j) \in \mathsf{BS}_n$ and $h \in \mathbb{N} \cup \{0\}$, let $\mathbf{\Delta}^{(h)}_{i..j} := \mathbf{M}^{(h)}_{i..j} - \mathbf{M}_{i..j}$. In the proof of Lemma 7.17 we saw that $\mathbf{\Delta}^{(k)}_{\ell..r}$ equals to

$$\sum_{i+j=k} \mathbf{\Delta}^{(i)}_{\ell..m}\mathbf{\Delta}^{(j)}_{m..r} - \sum_{i+j=k-1} \mathbf{\Delta}^{(i)}_{\ell..m}\mathbf{\Delta}^{(j)}_{m..r} + \mathbf{\Delta}^{(k)}_{\ell..m}\mathbf{M}_{m..r} + \mathbf{M}_{\ell..m}\mathbf{\Delta}^{(k)}_{m..r}.$$

The $O(n\varepsilon_0)^{k+1}$ bound in Lemma 7.17 comes from repeatedly applying sub-additivity and sub-multplicativity to the formula above. To get a better error bound, our first observation is that the last two error terms in the above formula combine nicely in the regular setting. That is, by induction hypothesis we can show that the second last error term $\mathbf{\Delta}^{(k)}_{\ell..m}\mathbf{M}_{m..r}$ is proportional to the amount of mixing from layer ℓ to layer m, and the last error term $\mathbf{M}_{\ell..m}\mathbf{\Delta}^{(k)}_{m..r}$ is proportional to the amount of mixing from layer m to layer r. Therefore, their sum is proportional to the total amount of mixing from layer ℓ to layer r. Furthermore, we observe that with a proper choice of parameters, the error terms in the first two summations ($\sum_{i+j=k} \mathbf{\Delta}^{(i)}_{\ell..m} \cdot \mathbf{\Delta}^{(j)}_{m..r}$ and $\sum_{i+j=k-1} \mathbf{\Delta}^{(i)}_{\ell..m} \cdot \mathbf{\Delta}^{(j)}_{m..r}$) are actually very small compared to the last two terms, and thus do not affect the total error too much.

Specifically, consider a sequence of error parameter $(\varepsilon^{(i)})_{i\geq 0}$ to be defined later, and suppose we already know that the "magnitude" of $\mathbf{\Delta}^{(i)}_{\ell..m}, \mathbf{\Delta}^{(i)}_{m..r}$ (which corresponds to the error of $\mathbf{M}_{\ell..m}$ and $\mathbf{M}_{m..r}$ that we haven't formally defined yet) is roughly bounded by $\varepsilon^{(i)}$ for every i. Our goal is to prove by induction that the magnitude of the new error matrix $\mathbf{\Delta}^{(k)}_{\ell..r}$ is also roughly bounded by $\varepsilon^{(k)}$. We ob-

serve that if we choose $\varepsilon^{(i)}$ as in the following lemma, the error terms in the first two summations sum up to roughly $\varepsilon^{(k)}/\log(n)$, which is much smaller than the target error $\varepsilon^{(k)}$, and thus does not affect the total error too much.

Lemma 8.6. *Let $\gamma < 1/2$, and define $\varepsilon^{(i)} = \frac{\gamma^{i+1}}{10\log(n)(i+1)^2}$. Then for every $k \in \mathbb{N}$ we have*

$$\sum_{i+j=k} \varepsilon^{(i)}\varepsilon^{(j)} + \sum_{i+j=k-1} \varepsilon^{(i)}\varepsilon^{(j)} \leq \varepsilon^{(k)}/\log(n).$$

Proof.

$$\begin{aligned}
\sum_{i+j\in\{k-1,k\}} \varepsilon^{(i)}\varepsilon^{(j)} &= \frac{(k+1)^2\varepsilon^{(k)}}{10\log(n)} \cdot \left(\sum_{i+j\in\{k-1,k\}} \frac{\gamma^{i+j-k+1}}{(i+1)^2(j+1)^2} \right) \\
&= \frac{(k+1)^2\varepsilon^{(k)}}{10\log(n)} \cdot \sum_{i+j\in\{k-1,k\}} \gamma^{i+j-k+1} \cdot \left(\frac{\frac{1}{(i+1)^2} + \frac{1}{(j+1)^2}}{(i+1)^2 + (j+1)^2} \right) \\
&\leq \frac{(k+1)^2\varepsilon^{(k)}}{10\log(n)} \sum_{i+j\in\{k-1,k\}} \gamma^{i+j-k+1} \cdot \left(\frac{\frac{1}{(i+1)^2} + \frac{1}{(j+1)^2}}{(i+j+2)^2/2} \right) \\
&\leq \frac{(k+1)^2\varepsilon^{(k)}}{10\log(n)} \sum_{i+j\in\{k-1,k\}} \gamma^{i+j-k+1} \cdot \left(\frac{\frac{1}{(i+1)^2} + \frac{1}{(j+1)^2}}{(k+1)^2/2} \right) \\
&\leq \frac{\varepsilon^{(k)}}{10\log(n)} \cdot \left(4(1+\gamma) \sum_{i=0}^{k} \frac{1}{(i+1)^2} \right) \\
&\leq \frac{\varepsilon^{(k)}}{\log(n)}.
\end{aligned}$$

□

With the choice of parameters above, we can prove that the error of the "level-k approximation" $\mathbf{M}^{(k)}_{\ell..r}$ only grows by a factor of $(1 + 1/\log(n))$ after each level of recursion. After $\log(n)$ levels of recursion, the error only grows by a constant factor. Therefore, we can choose the "base-case error" $\varepsilon^{(0)}$ to be as small as $O(1/\log(n))$. This allows us to choose base cases with small seed length or space complexity. For the proof of Theorem 8.3, we choose the base case to be the BRRY PRG [BRRY14]

with error $2^{-\sqrt{\log(1/\varepsilon)}}$. For the proof of Theorem 8.2 the base cases are generated using derandomized squaring [RV05, APP+23].

8.3 WPRG for Regular ROBPs

In this section we prove Theorem 8.3. In this section we use the notation that for any alphabet Σ and string $x \in \Sigma^*$, we use $x_{[i]}$ to denote the i-th symbol of x and $x_{[\le i]}$ to denote the prefix of x of length i.

Fix a regular (n, w)-ROBP and its corresponding transition matrices $\mathbf{M}_{[n]}$. Using the matrix notation, the weight defined in [BRRY14] can be written as follows.[31]

Definition 8.7. *For every vector $y \in \mathbb{R}^w$ and every $i \in [n]$, define the* layer-i weight on y *as*

$$W(i, y) := \sum_{u \in [w]} \sum_{b \in \{0,1\}} \left| (\mathbf{M}_i y)[u] - y[B_i(u, b)] \right|.$$

For every $0 \le \ell < r \le n$, the total weight between layer ℓ and r on y is defined as

$$W(\ell, r, y) := \sum_{i=\ell+1}^{r} W(i, \mathbf{M}_{i..r} y).^{32}$$

Remark 8.8. *To interpret $W(\ell, r, y)$ with the original description in [BRRY14], consider the graph view of ROBPs, and consider y to be the values on the nodes in layer r. Then for every $i \le r$, $\mathbf{M}_{i..r} y$ corresponds to the values on layer i. Observe that each term in the definition of $W(i, \mathbf{M}_{i..r} y)$ corresponds to the "weight" on an edge between layer $i-1$ and i. In consequence, $W(\ell, r, y)$ corresponds to the total weight of the* sub-program between layer ℓ and r *(i.e., the ROBP specified by transition functions $(B_{\ell+1}, \ldots, B_r)$).*

[31] The original results in [BRRY14] only consider $y \in [0, 1]^w$, but one can easily generalize them to $\mathbb{R}^w$ by shifting and scaling.

[32] For the degenerate case $i = r$, let $\mathbf{M}_{r..r}$ denote the identity matrix.

The following identity is straightforward by definition:

Fact 8.9. *For every $0 \leq \ell < m < r \leq n$ and every vector $y \in \mathbb{R}^w$, $W(\ell, r, y) = W(\ell, m, \mathbf{M}_{m..r}y) + W(m, r, y)$. This also implies $\max(W(\ell, m, \mathbf{M}_{m..r}y), W(m, r, y)) \leq W(\ell, r, y)$.*

Given the definition of weight, the main results in [BRRY14] imply the following lemmas. Note that Lemma 8.10 is the only place where regularity is required in this section, and Lemma 8.11 does not require regularity.

Lemma 8.10. *For every $\ell < r$ and every vector $y \in \mathbb{R}^w$, $W(\ell, r, y) \leq w^2 \|y\|_\infty$.*

Lemma 8.11. *For every $\delta > 0$, there exists an explicit function $G_0 : \{0,1\}^{d_0} \to \{0,1\}^n$ s.t.,*

$$\left\| \left(\mathop{\mathbb{E}}_{s \sim \{0,1\}^{d_0}} [\mathbf{M}_{\ell..r}(G_0(s))] - \mathbf{M}_{\ell..r} \right) y \right\|_\infty \leq \delta W(0, n, y).$$

Using the fact that any ROBP of length $r - \ell \leq n$ can be extended to be of length n by adding dummy identity transitions, one can get the following corollary.

Corollary 8.12. *For every $\delta > 0$, there exists an explicit PRG $G_0 : \{0,1\}^{d_0} \to \{0,1\}^n$ s.t. for every $0 \leq \ell < r \leq n$,*

$$\left\| \left(\mathop{\mathbb{E}}_{s \sim \{0,1\}^{d_0}} \left[\mathbf{M}_{\ell..r}(G_0(s)_{[\leq r-\ell]})\right] - \mathbf{M}_{\ell..r} \right) y \right\|_\infty \leq \delta W(\ell, r, y),$$

In addition, the seed length is $d_0 = O\left(\log(n)\left(\log\log(n) + \log(w/\delta)\right)\right)$.

Now define $W^* := w^2$, which by Lemma 8.10 implies $W^* \geq \max_{y:\|y\|_\infty=1}(W(0, n, y))$. To simplify notation, we define *weight approximation* as follows.

Definition 8.13. *For every $0 \leq \ell < r \leq n$, we say $\widetilde{\mathbf{M}_{\ell..r}}$ is a ε_0-weight approximation of $\mathbf{M}_{\ell..r}$ if*

$$\forall y \in \mathbb{R}^w \quad \left\| \left(\widetilde{\mathbf{M}_{\ell..r}} - \mathbf{M}_{\ell.r} \right) y \right\|_\infty \leq \varepsilon_0 \cdot \frac{W(\ell, r, y)}{W^*}.$$

Note that $\widetilde{\mathbf{M}_{\ell..r}}$ being a δ-weight approximation of $\mathbf{M}_{\ell..r}$ also implies $\left\|\widetilde{\mathbf{M}_{\ell..r}} - \mathbf{M}_{\ell..r}\right\|_\infty \leq \delta$. Now fix a parameter $\gamma > 0$ to be specified later, and define $\varepsilon^{(i)} = \frac{\gamma^{i+1}}{10(i+1)^2 \log(n)}$ as in Lemma 8.6. Let G_0 be the PRG in Corollary 8.12 with parameter $\delta = \varepsilon^{(0)}/(3W^*)$, and for every $(\ell, r) \in \mathsf{BS}_n$ such that $r - \ell > 1$, define

$$\mathbf{M}^{(0)}_{\ell..r} := \mathop{\mathbb{E}}_{s \sim \{0,1\}^{d_0}} \left[\mathbf{M}_{\ell..r}(G_0(s)_{[\leq r-\ell]})\right],$$

which is a $(\varepsilon^{(0)}/3)$-weight approximation by Corollary 8.12. Then define $\mathbf{M}^{(k)}_{\ell..r}$ recursively as in Claim 8.4. The following is our main lemma for proving Theorem 8.3:

Lemma 8.14 (main)**.** *For every $k \in \mathbb{N}$, every $y \in \mathbb{R}^w$ and every $(\ell, r) \in \mathsf{BS}_n$, $\mathbf{M}^{(k)}_{\ell..r}$ is a $C_t\varepsilon^{(k)}$-weight approximation of $\mathbf{M}_{\ell..r}$, where $t = \log(r - \ell)$ and $C_t = (1 + 1/\log(n))^t/3$.*

Proof. We prove the lemma by induction over t and k. The first base case $t = 0$ is trivial since $\mathbf{M}^{(k)}_{\ell..r} = \mathbf{M}_{\ell..r}$. The second base case $k = 0$ is also true by definition. For the general case, first we note that the lemma also implies $\left\|\mathbf{\Delta}^{(k)}_{\ell.r}\right\|_\infty \leq C_t\varepsilon^{(k)} \leq \varepsilon^{(k)}$. Then observe that by Lemma 7.16 and sub-additivity/sub-multiplicativity of infinity norm, we have

$$\begin{aligned}
&\left\|\mathbf{\Delta}^{(k)}_{\ell..r} y\right\|_\infty \\
&\leq \left(\sum_{i+j\in\{k-1,k\}} \left\|\mathbf{\Delta}^{(i)}_{\ell..m}\mathbf{\Delta}^{(j)}_{m..r} y\right\|_\infty\right) + \left\|\mathbf{M}_{\ell..m}\mathbf{\Delta}^{(k)}_{m..r} y\right\|_\infty + \left\|\mathbf{\Delta}^{(k)}_{\ell..m}\mathbf{M}_{m..r} y\right\|_\infty \\
&\leq \left(\sum_{i+j\in\{k-1,k\}} \left\|\mathbf{\Delta}^{(i)}_{\ell..m}\right\|_\infty \left\|\mathbf{\Delta}^{(j)}_{m..r} y\right\|_\infty\right) + \|\mathbf{M}_{\ell..m}\|_\infty \left\|\mathbf{\Delta}^{(k)}_{m..r} y\right\|_\infty + \left\|\mathbf{\Delta}^{(k)}_{\ell..m}\mathbf{M}_{m..r} y\right\|_\infty \\
&\leq \left(\sum_{i+j\in\{k-1,k\}} C^2_{t-1}\varepsilon^{(i)}\varepsilon^{(j)} \cdot \frac{W(m,r,y)}{W^*}\right) \\
&\quad + C_{t-1}\varepsilon^{(k)} \cdot \frac{W(m,r,y) + W(\ell,m,\mathbf{M}_{m..r}y)}{W^*} && \text{(by induction)} \\
&\leq C_t\varepsilon^{(k)} \cdot \frac{W(\ell,r,y)}{W^*}. && \text{(by Lemma 8.6 and Fact 8.9)}
\end{aligned}$$

In other words, $\mathbf{M}^{(k)}_{\ell..r}$ is a $C_t\varepsilon^{(k)}$-weight approximation of $\mathbf{M}_{\ell..r}$. □

Lemma 8.14 shows that $\mathbf{M}^{(k)}_{0..n}$ is a $\varepsilon^{(k)}$-weight approximation, which also implies $\left\|\mathbf{M}^{(k)}_{0..n} - \mathbf{M}_{0..n}\right\|_\infty \le \varepsilon^{(k)}$. It remains to construct a WPRG that actually "implements" $\mathbf{M}^{(k)}_{0..n}$. This step is rather standard and is essentially the same as the corresponding step in [CHL+23]: to get the seed length as claimed in Theorem 8.3, we need to further "derandomize" $\mathbf{M}^{(k)}_{0..n}$ using the technique in [CDR+21, PV21].[33] In short, we expand the recursive formula for $\mathbf{M}^{(k)}_{0..n}$ and get an "error reduction polynomial" over matrices $\mathbf{M}^{(0)}_{i..j}$. One can show that there are at most $K = n^{O(k)}$ terms in the polynomial, and each term has at most $h = k\log(n)$ factors. Then we can use the INW generator [INW94] for length h and width w to approximate each term with error $\varepsilon/2K$, which gives us a $\varepsilon/2$-approximation to $\mathbf{M}^{(k)}_{0..n}$. We discuss the details in Section 8.3.1.

Remark 8.15. *Note that our construction and proof also works for "small-weight ROBPs" in general, if we define $W^* = \max_{B,y:\|y\|_\infty=1}(W(0,n,y))$ instead. This only costs additional $O(\log(n)\log(W^*))$ bits of seed length. We note that this generality is important in some applications, such as the WPRG for width-3 ROBP in [CHL+23].*

8.3.1 Final WPRG Construction

To simplify notation, we assume w.l.o.g. that the first output bit of G_0 is unbiased, i.e., $\Pr_{s\in\{0,1\}^{d_0}}\left[G_0(d_0)_{[1]} = 1\right] = 1/2$.[34] Then we can merge the two different base cases by defining $\mathbf{M}^{(0)}_{r-1..r} := \mathbf{M}_r = \mathbf{M}^{(k)}_{r-1..r} = \mathbb{E}_{s\in\{0,1\}^{d_0}}\left[\mathbf{M}_{r-1..r}(G_0(s)_{[\le 1]})\right]$. Now consider the following notation.

Definition 8.16. *For every $0 \le \ell < r \le n$, let $\mathsf{IS}_{\ell..r}$ denote the set of increasing sequences $\mathsf{sq} = (i_0, i_1 \ldots, i_h)$ s.t. $\ell = i_0 < i_1 < \ldots < i_h = r$. We say h is the* length *of* sq. *For every*

[33] One may use the sampler-based derandomization in Chapter 7 instead, but it requires more care to argue that the sampler error is also proportional to the weight.

[34] To get such a PRG, we can simply take a PRG G'_0 from [BRRY14] with $(n-1)$-bit output and define $G_0(b \circ s) = b \circ G'_0(s)$, where b is the first bit.

$\mathsf{sq} \in \mathsf{IS}_{\ell..r}$, *define*

$$\mathbf{M}^{(0)}_{\mathsf{sq}} := \prod_{j=1}^{h} \mathbf{M}^{(0)}_{i_{j-1}..i_j}.$$

Given the notation above, we get the following lemma regarding the expansion of $\mathbf{M}^{(k)}_{0..n}$, which is not hard to prove by induction.

Lemma 8.17. *For every $k \in \mathbb{N}$ and every $(\ell, r) \in \mathsf{BS}_n$, there is a (multi)set $S \subseteq \mathsf{IS}_{\ell..r} \times \{-1, +1\}$ which satisfies that*

- $\mathbf{M}^{(k)}_{\ell..r} = \sum_{(\mathsf{sq},\sigma)\in S} \sigma \mathbf{M}^{(0)}_{\mathsf{sq}}$.
- $|S| \leq (r-\ell)^{2k}$
- *For every $(\mathsf{sq}, \sigma) \in S$, the length of sq is at most $k \log(r-\ell) + 1$*

Proof. For $\mathsf{sq}_\ell = (i_0, \ldots, i_{h_\ell}) \in \mathsf{IS}_{\ell..m}$ and $\mathsf{sq}_r = (j_0, \ldots, j_{h_r}) \in \mathsf{IS}_{m..r}$, define

$$\mathsf{sq}_\ell \| \mathsf{sq}_r = (i_0, \ldots, i_{h_\ell} = j_0, \ldots, j_{h_r}).$$

Observe that $\mathsf{sq}_\ell || \mathsf{sq}_r$ has length $h_\ell + h_r$, and that $\mathbf{M}^{(0)}_{\mathsf{sq}_\ell || \mathsf{sq}_r} = \mathbf{M}^{(0)}_{\mathsf{sq}_\ell} \cdot \mathbf{M}^{(0)}_{\mathsf{sq}_r}$.

With the notation above, we prove the lemma by induction over $t := \log(r-\ell)$ and k. The base cases $t = 0$ or $k = 0$ are trivial. Let $S^{(i)}_\ell, S^{(j)}_r$ denote the corresponding sets of $\mathbf{M}^{(i)}_{\ell..m}$ and $\mathbf{M}^{(j)}_{m..r}$ respectively. By distributive law, we get

$$\mathbf{M}^{(k)}_{\ell..r} = \sum_{b\in\{0,1\}} \sum_{i+j=k-b} \sum_{(\mathsf{sq}_\ell,\sigma)\in S^{(i)}_\ell} \sum_{(\mathsf{sq}_\mathsf{r},\sigma)\in S^{(j)}_r} (-1)^b \sigma_\ell \sigma_r \mathbf{M}^{(0)}_{\mathsf{sq}_\ell \| \mathsf{sq}_r}.$$

Therefore, we can define

$$S = \bigcup_{b\in\{0,1\}^{0,1}, i+j=k-b} \{(\mathsf{sq}_\ell \| \mathsf{sq}_r, (-1)^b \sigma_\ell \sigma_r) : (\mathsf{sq}_\ell, \sigma) \in S^{(i)}_\ell \wedge (\mathsf{sq}_\mathsf{r}, \sigma) \in S^{(j)}_r\},$$

which satisfies the first condition. Based on the induction hypothesis, the length of each $\mathsf{sq}_\ell \| \mathsf{sq}_r$ is at most $(i+j)(t-1)+2 \le k(t-1)+2 \le kt+1$, and the size of S is at most

$$\sum_{i+j \in \{k-1,k\}} \left|S_\ell^{(i)}\right| \left|S_r^{(j)}\right| \le (2k+1)2^{2k(t-1)} \le 2^{2kt}.$$

□

In addition, we would need to derandomize each term $\mathbf{M}_{\mathsf{sq}}^{(0)}$ using the following matrix view of INW generator [INW94], which can be found in, e.g., [BCG20]:

Lemma 8.18. *Let Σ be a finite set of symbols. Suppose for every $i \in [h]$, there is a matrix-valued function $\mathbf{A}_i : \Sigma \to \mathbb{R}^{w \times w}$ which on every input in Σ outputs a stochastic matrix. Then for every $\varepsilon_{\mathrm{INW}} > 0$ there exists an explicit function $G_{\mathrm{INW}} : \{0,1\}^d \to \Sigma^h$ such that*

$$\left\| \mathop{\mathbb{E}}_{s \in \{0,1\}^d} \left[\prod_{i=1}^{h} \mathbf{A}_i(G_{\mathrm{INW}}(s)_{[i]}) \right] - \prod_{i=1}^{h} \mathop{\mathbb{E}}_{x \in \Sigma} [\mathbf{A}_i(x)] \right\|_\infty \le \varepsilon_{\mathrm{INW}},$$

and $d = O(\log|\Sigma| + \log(h)\log(hw/\varepsilon_{\mathrm{INW}}))$.

Now we are ready to prove Theorem 8.3.

Proof of Theorem 8.3. Let $S \subseteq \mathsf{IS}_{0..n} \times [-1,1]$ be the set defined in Lemma 8.17 s.t. $\mathbf{M}_{0..n}^{(k)} = \sum_{(\mathsf{sq},\sigma)\in S} \sigma \mathbf{M}_{\mathsf{sq}}^{(0)}$. Without loss of generality we can assume that S has size exactly $2^{2k\log(n)}$ by adding dummy sequences with weight $\sigma = 0$. In addition, note that there is a enumeration function $E_S : \{0,1\}^{2k\log(n)} \to S$ that can be implemented in space $O(\log(k)\log(n))$ following the recursive formula in Claim 8.4.[35] Then consider $G_{\mathrm{INW}} : \{0,1\}^{d_{\mathrm{INW}}} \to \Sigma^h$ in Lemma 8.18 with $\Sigma = \{0,1\}^{d_0}, h = k\log(n)+1$ and $\varepsilon_{\mathrm{INW}} = \varepsilon/(2|S|)$, then define $d = 2k\log(n) + d_{\mathrm{INW}}$. The final WPRG construction $(\rho, G) : \{0,1\}^d \to \mathbb{R} \times \{0,1\}^n$ is as follows. On any input s,

[35] That is, we use the first $\lceil \log(2k+1) \rceil$ bits as index to determine a term in the recursive formula, then discard these $\lceil \log(2k+1) \rceil$ bits and recurse. If there's any undefined index, simply return a dummy sequence with weight 0.

1. Parse s as $(s_{\text{enum}}, s_{\text{INW}}) \in \{0,1\}^{2k\log(n)} \times \{0,1\}^{d_{\text{INW}}}$
2. Define $((i_0, i_1, \ldots, i_h), \sigma) := E_S(s_{\text{enum}})$.
3. For $j \in [h]$, define $r_j := G_0(G_{\text{INW}}(y)_{[j]}) \in \{0,1\}^n$.
4. Output $(\rho(s), G(s)) := (\sigma|S|, (r_1)_{[\le i_1 - i_0]} \circ (r_2)_{[\le i_2 - i_1]} \circ \ldots \circ (r_h)_{[\le i_h - i_{h-1}]})$.

Next we prove the correctness of G. Observe that

$$\mathop{\mathbb{E}}_{s}\left[\rho(s)\mathbf{M}_{0..n}(G(s))\right] = \sum_{((i_0,\ldots,i_h),\sigma)\in S} \sigma \cdot \mathop{\mathbb{E}}_{s_{INW}}\left[\prod_{j=1}^{h} \mathbf{M}_{i_{j-1}..i_j}(G_0(G_{\text{INW}}(s_{\text{INW}})_{[j]}))\right].$$

For every term in the above equation, consider the matrix-valued functions $\mathbf{A}_j : \{0,1\}^{d_0} \to \mathbb{R}^{w\times w}$ s.t. $\mathbf{A}_j(r) = \mathbf{M}_{i_{j-1}..i_j}(G_0(r)_{[\le i_j - i_{j-1}]})$. Note that $\mathbb{E}_r\left[\mathbf{A}_j(r)\right] = \mathbf{M}^{(0)}_{i_{j-1}..i_j}$. Then by Lemma 8.18 we have

$$\left\| \mathop{\mathbb{E}}_{s_{INW}}\left[\prod_{j=1}^{h} \mathbf{M}_{i_{j-1}..i_j}(G_{\text{INW}}(s_{\text{INW}})_j)\right] - \mathbf{M}^{(0)}_{(i_0,i_1,\ldots,i_h)} \right\|_\infty \le \varepsilon/(2|S|),$$

which by the sub-additivity of $\|\cdot\|_\infty$ implies

$$\left\| \mathop{\mathbb{E}}_{s}\left[\rho(s)\mathbf{M}_{0..n}(G(s))\right] - \mathbf{M}^{(k)}_{0..n} \right\|_\infty \le \varepsilon/2.$$

We pick suitable γ, k (to be specified later) so that $\varepsilon^{(k)} \le \varepsilon/2$. Then by Lemma 8.14 we have

$$\left\| \mathop{\mathbb{E}}_{s}\left[\rho(s)\mathbf{M}_{0..n}(G(s))\right] - \mathbf{M}^{(k)}_{0..n} \right\|_\infty \le \varepsilon.$$

Then consider the vectors $v_{\mathsf{st}}, v_{\mathsf{ed}} \in \mathbb{R}^w$ corresponding to the start and end states as discussed in Section 7.2. Observe that $\|v_{\mathsf{st}}\|_1 = 1$ and $\|v_{\mathsf{ed}}\|_\infty \le 1$ by definition. Therefore we have

$$\begin{aligned}
&\left| \mathop{\mathbb{E}}_{s\in\{0,1\}^d}\left[\rho(s)B(G(s))\right] - \mathop{\mathbb{E}}_{x\in\{0,1\}^n}\left[B(x)\right] \right| \\
&= \left| v_{\mathsf{st}}^\top \left(\mathop{\mathbb{E}}_{s}\left[\rho(s)\mathbf{M}_{0..n}(G(s))\right] - \mathop{\mathbb{E}}_{x}\left[\mathbf{M}_{0..n}(x)\right] \right) v_{\mathsf{ed}} \right| \\
&\le \|v_{\mathsf{st}}\|_1 \cdot \left\| \mathop{\mathbb{E}}_{s}\left[\rho(s)\mathbf{M}_{0..n}(G(s))\right] - \mathbf{M}_{0..n} \right\|_\infty \cdot \|v_{\mathsf{ed}}\|_\infty \\
&\le \varepsilon.
\end{aligned}$$

Finally we analyze the seed length with an unspecified parameter $0 < \gamma < 1/\log(n)$. Take k to be the minimum integer s.t. $\varepsilon^{(k)} \leq \varepsilon/2$. Observe that $\log(1/\varepsilon^{(0)}) = O(1/\gamma)$ and $k = O(\log(1/\varepsilon)/\log(1/\gamma))$. This implies

$$d = d_0 + O(\log(h)\log(hw/\varepsilon) + k\log(n)) = d_0 + \tilde{O}(\log(w/\varepsilon) + k\log(n)).$$

By Corollary 8.12, we have

$$d_0 = O(\log(n)\log(wW^*/\gamma) + \log(n)\log\log(n)) = \tilde{O}(\log(n)\log(w/\gamma)).$$

Therefore,

$$d = \tilde{O}\left(\log(n)\log(w/\gamma) + \frac{\log(n)\log(1/\varepsilon)}{\log(1/\gamma)} + \log(1/\varepsilon)\right).$$

Taking $\gamma = 2^{-\sqrt{\log(n)}}$, we get

$$d = \tilde{O}\left(\log(n)\left(\log(w) + \sqrt{\log(1/\varepsilon)}\right) + \log(1/\varepsilon)\right).$$

Observe that the space complexity is $O(d_0 + \log(k)\log(n) + d_{\mathrm{INW}}) = O(d)$. Therefore the WPRG is explicit. □

8.4 Non-black-box Derandomization for Regular ROBPs

In this section we prove Theorem 8.2. Inspired by [CHL+23], we use the notion of SV approximation [APP+23] to capture the "amount of mixing". To simplify notation, we define the function $D : \mathbb{R}^{w\times w} \times \mathbb{R}^w \to \mathbb{R}$ to be $D(\mathbf{A}, y) := \|y\|^2 - \|\mathbf{A}y\|^2$, which plays the same role as the weight measure in the proof of Theorem 8.3. The following fact is straightforward from the definition.

Fact 8.19. $D(\mathbf{B}, y) + D(\mathbf{A}, \mathbf{B}y) = D(\mathbf{AB}, y)$.

The notion of SV approximation is defined as follows.

Definition 8.20 (SV approximation [APP+23])**.** *Let $\mathbf{W} \in \mathbb{R}^{w \times w}$ be a doubly stochastic matrix. We say $\widetilde{\mathbf{W}}$ is a ε-SV approximation of W if for every $x, y \in \mathbb{R}^w$,*

$$\left| x^\top (\widetilde{\mathbf{W}} - \mathbf{W}) y \right| \leq \varepsilon \cdot \left(\frac{D(\mathbf{W}^\top, x) + D(\mathbf{W}, y)}{2} \right).$$

Equivalently, for every $x, y \in \mathbb{R}^w$,

$$\left| x^\top (\widetilde{\mathbf{W}} - \mathbf{W}) y \right| \leq \varepsilon \cdot \left(\sqrt{D(\mathbf{W}^\top, x) \cdot D(\mathbf{W}, y)} \right).$$

The proof of Theorem 8.2 is very similar to our proof of Theorem 8.3 in the previous section. First we also need a base case for the different approximation notion. As proved in [APP+23, CHL+23], there is a space-efficient implementation of SV approximation of random walk matrices based on derandomized squaring [RV05]:

Lemma 8.21 ([APP+23, CHL+23])**.** *For every $(\ell, r) \in \mathsf{BS}_n$, there is an algorithm that computes a δ-SV approximation of $\mathbf{M}_{\ell..r}$ in space $\tilde{O}(\log(nw)\log(1/\delta))$. Further, each entry of this approximation matrix has bit length at most $O(\log(n)\log(1/\delta))$.*

We also need the following simple lemma, which can be found in [CHL+23]. We include its (short) proof for completeness.

Lemma 8.22. *Suppose $\widetilde{\mathbf{W}}$ is a δ-SV-approximation of $\mathbf{W}$, and let $\mathbf{\Delta} = \widetilde{\mathbf{W}} - \mathbf{W}$. Then*

$$\|\mathbf{\Delta} y\|_2 \leq \delta \sqrt{D(\mathbf{W}, y)}.$$

Proof. Observe that

$$\|\mathbf{\Delta} y\|_2^2 = (\mathbf{\Delta} y)^\top \mathbf{\Delta} y \leq \delta \sqrt{D(\mathbf{W}, y) \cdot (\|\mathbf{\Delta} y\|_2^2 - \|\mathbf{W}^\top \mathbf{\Delta} y\|_2^2)} \leq \delta \sqrt{D(\mathbf{W}, y)} \cdot \|\mathbf{\Delta} y\|.$$

□

Now for every $(\ell, r) \in \mathsf{BS}_n$, define $\mathbf{M}^{(0)}_{\ell..r}$ to be a $(\varepsilon^{(0)}/3)$-SV approximation of $\mathbf{M}_{\ell..r}$, and define $\mathbf{M}^{(k)}_{\ell..r}$ using the recursion (Claim 8.4). We prove the following lemma which is analogous to Lemma 8.14:

Lemma 8.23 (main). *For every $k \in \mathbb{N}$ and every $(\ell, r) \in \mathsf{BS}_n$, $\mathbf{M}^{(k)}_{\ell..r}$ is a $C_t\varepsilon^{(k)}$-SV approximation of $\mathbf{M}_{\ell..r}$, where $t = \log(r-\ell)$ and $C_t = (1+1/\log(n))^t/3$.*

Proof. We again prove the lemma by induction. The base cases $t = 0$ or $k = 0$ are trivial by definition. For the inductive case, for every $x, y \in \mathbb{R}^w$ we have

$$x^\top \mathbf{\Delta}^{(k)}_{\ell..r} y = \sum_{i+j=k} x^\top \mathbf{\Delta}^{(i)}_{\ell..m} \mathbf{\Delta}^{(j)}_{m..r} y - \sum_{i+j=k-1} x^\top \mathbf{\Delta}^{(i)}_{\ell..m} \mathbf{\Delta}^{(j)}_{m..r} y + x^\top \mathbf{M}_{\ell..m} \mathbf{\Delta}^{(k)}_{m..r} y + x^\top \mathbf{\Delta}^{(k)}_{\ell..m} \mathbf{M}_{m..r} y. \tag{8.1}$$

To bound the first two summations in Equation (8.1), observe that

$$\begin{aligned}
&\sum_{i+j=k} x^\top \mathbf{\Delta}^{(i)}_{\ell..m} \mathbf{\Delta}^{(j)}_{m..r} y - \sum_{i+j=k-1} x^\top \mathbf{\Delta}^{(i)}_{\ell..m} \mathbf{\Delta}^{(j)}_{m..r} y \\
&\le \sum_{i+j\in\{k-1,k\}} \left\| (\mathbf{\Delta}^{(i)}_{\ell..m})^\top x \right\|_2 \left\| \mathbf{\Delta}^{(j)}_{m..r} y \right\|_2 && \text{(Cauchy-Schwarz)} \\
&\le C^2_{t-1} \sum_{i+j\in\{k-1,k\}} \varepsilon^{(i)}\varepsilon^{(j)} \sqrt{D\left(\mathbf{M}^\top_{\ell..m}, x\right) D\left(\mathbf{M}_{m..r}, y\right)} && \text{(Lemma 8.22)} \\
&\le C^2_{t-1} \cdot \frac{\varepsilon^{(k)}}{\log(n)} \cdot \frac{D\left(\mathbf{M}^\top_{\ell..m}, x\right) + D\left(\mathbf{M}_{m..r}, y\right)}{2} \\
& && \text{(by Lemma 8.6 and AM-GM)} \\
&\le C_{t-1} \cdot \frac{\varepsilon^{(k)}}{\log(n)} \cdot \frac{D\left(\mathbf{M}^\top_{\ell..r}, x\right) + D\left(\mathbf{M}_{\ell..r}, y\right)}{2} \\
& && \text{(since } C_{t-1} \le 1 \text{ and } \|\mathbf{M}_{\ell..m}\|_2, \left\|\mathbf{M}^\top_{m..r}\right\|_2 \le 1\text{)}
\end{aligned}$$

To bound the last two terms in Equation (8.1), observe that

$$\begin{aligned}
&x^\top \mathbf{M}_{\ell..m} \mathbf{\Delta}^{(k)}_{m..r} y + x^\top \mathbf{\Delta}^{(k)}_{\ell..m} \mathbf{M}_{m..r} y \\
&\le C_{t-1} \cdot \varepsilon^{(k)} \cdot \frac{D(\mathbf{M}_{m..r}, y) + D(\mathbf{M}^\top_{m..r}, \mathbf{M}^\top_{\ell..m} x) + D(\mathbf{M}_{\ell..m}, \mathbf{M}_{m..r} y) + D(\mathbf{M}^\top_{\ell..m}, x)}{2} \\
&= C_{t-1} \cdot \varepsilon^{(k)} \cdot \frac{D(\mathbf{M}_{\ell..r}, y) + D(\mathbf{M}^\top_{\ell..r}, x)}{2}. && \text{(Fact 8.9)}
\end{aligned}$$

By summing up the two inequalities, we can conclude that

$$x^\top \mathbf{\Delta}^{(k)}_{\ell..r} y \le C_t \cdot \varepsilon^{(k)} \cdot \frac{D(\mathbf{M}_{\ell..r}, y) + D(\mathbf{M}^\top_{\ell..r}, x)}{2}.$$

Because negating y does not change the bound above, we get

$$\left|x^\top(\mathbf{M}^{(k)}_{\ell..r} - \mathbf{M}_{\ell..r})y\right| \le C_t \cdot \varepsilon^{(k)} \cdot \frac{D(\mathbf{M}_{\ell..r}, y) + D(\mathbf{M}^\top_{\ell..r}, x)}{2},$$

i.e., $\mathbf{M}^{(k)}_{\ell..r}$ is a $C_t\varepsilon^{(k)}$-SV approximation of $\mathbf{M}_{\ell..r}$. □

Finally, let $\gamma = 1/\log(n)$ and $k = O(\log(1/\varepsilon)/\log(1/\gamma))$ be the minimum integer s.t. $\varepsilon^{(k)} \le \varepsilon$. We claim that $\mathbf{M}^{(k)}_{0..n}$ can be implemented in space $\tilde{O}(\log(k)\log(nw))$, which implies the following result:

Theorem 8.24. *There is an algorithm which can compute an ε-SV approximation of the random walk matrix $\mathbf{M}_{0..n}$ in space $\tilde{O}(\log(nw)\log\log(1/\varepsilon))$.*

Note that *Theorem* 8.2 is also a direct corollary of this theorem:

Proof of Theorem 8.2. Compute a $(\varepsilon/\sqrt{w})$-SV approximation of $\mathbf{M}_{0..n}$, denoted by $\widetilde{\mathbf{M}_{0..n}}$. By Theorem 8.24 this takes space $\tilde{O}(\log(nw)\log\log(1/\varepsilon))$. Then consider $v_{\mathsf{st}}, v_{\mathsf{ed}}$ as discussed in Section 7.2, and output $v_{\mathsf{st}}^\top\widetilde{\mathbf{M}_{0..n}}v_{\mathsf{ed}}$. To prove the correctness, recall that $v_{\mathsf{st}}^\top\mathbf{M}_{0..n}v_{\mathsf{ed}} = \mathbb{E}_x[B(x)]$, which implies

$$\left|v_{\mathsf{st}}^\top\widetilde{\mathbf{M}_{0..n}}v_{\mathsf{ed}} - \mathop{\mathbb{E}}_{x\in\{0,1\}^n}[B(x)]\right| = \left|v_{\mathsf{st}}^\top(\widetilde{\mathbf{M}_{0..n}} - \mathbf{M}_{0..n})v_{\mathsf{ed}}\right| \le \varepsilon/\sqrt{w}\,\|v_{\mathsf{st}}\|\,\|v_{\mathsf{ed}}\| \le \varepsilon$$

by the fact that $\|v_{\mathsf{st}}\| = 1$ and $\|v_{\mathsf{ed}}\| \le \sqrt{w}$. □

Remark 8.25. *Note that $\mathbf{M}^{(k)}_{0..n}$ does not satisfy the original definition of SV approximation in [APP+23] because it is not necessarily doubly stochastic. While every row and column in $\mathbf{M}^{(k)}_{0..n}$ does sum up to 1, some of its entries might be negative.*

8.4.1 Space-efficient Implementation

Finally we prove that $\mathbf{M}^{(k)}_{0..n}$ can be implemented in space $\tilde{O}(\log(k)\log(nw))$. Note that a naive implementation of the recursion in Claim 8.4 takes at least

$O(\log(n)\log(wk))$ space. Furthermore, we cannot naively enumerate each term of $\mathbf{M}^{(k)}_{0..n}$ in its expansion (Lemma 8.17) either, because there are $n^{O(k)}$ terms in total, which takes at least $O(k\log(n))$ bits to enumerate.

To reduce the space complexity, we will compute $\mathbf{M}_{0..n}$ with a different recursive formula. The intuition of the new recursion is as follows. First observe that each term in the expansion of $\mathbf{M}^{(k)}_{0..n}$ corresponds to a way to put k balls into $2n-1$ bins (indexed by $[2n-1]$), under the constraint that each odd-indexed bin contains at most one ball. To see why this is the case, observe that the original recursion corresponds to the following way to recursively enumerate all the ball-to-bin combinations: first we put $b \in \{0,1\}$ ball in the middle bin (which corresponds to the sign $(-1)^b$), then choose i, j s.t. $i + j = k - b$, and then recursively put i balls in the left $(n-1)$ bins (which corresponds to $\mathbf{M}^{(i)}_{0..n/2}$) and j balls in the right $(n-1)$ bins (which corresponds to $\mathbf{M}^{(j)}_{n/2..n}$).

Then observe that there is a different way to enumerate all the combinations with only $\lceil\log(k)\rceil$ levels of recursion as follows. First decide where the h-th ball is located, where $h = \lceil k/2\rceil$. If it is in an even-indexed bin, also decide how many balls are on the left and how many balls are on the right. Otherwise, there can be only one ball in the selected odd-indexed bin, and the numbers of balls on the left and right are fixed. Then for each choice, recursively enumerate the combinations on the left and right respectively. We claim that there is a corresponding recursive formula for $\mathbf{M}^{(k)}_{0..n}$ which can be implemented in only $\tilde{O}(\log(k)\log(nw))$ space.

To define this recursive formula, first we generalize the definition of $\mathbf{M}^{(k)}_{\ell..r}$ to $(\ell, r) \notin \mathsf{BS}_n$. For any (ℓ, r) s.t. $0 \le \ell < r \le n$, define $\mathsf{LCA}(\ell, r)$ as follows. Let t be the largest integer such that there exists a multiple of 2^t in the range (ℓ, r).

Then we define $\mathsf{LCA}(\ell, r)$ to be the unique multiple of 2^t in (ℓ, r).[36] Observe that for $(\ell, r) \in \mathsf{BS}_n$ s.t. $r - \ell > 1$, $\mathsf{LCA}(\ell, r) = (\ell + r)/2$. Therefore, we can generalize the recursion in Claim 8.4 to any $r - \ell > 1$ by defining $m = \mathsf{LCA}(\ell, r)$. We also generalize the same recursion to $k = 0, (\ell, r) \notin \mathsf{BS}_n$, so that $\mathbf{M}^{(0)}_{\ell..r} = \mathbf{M}^{(0)}_{\ell..m}\mathbf{M}^{(0)}_{m..r}$.[37] For the degenerate case $\ell = r$ we define $\mathbf{M}^{(k)}_{\ell..r}$ to be the identity matrix. Next we want to prove the following identity which naturally gives a recursive algorithm for $\mathbf{M}_{0..n}$. This identity is essentially the recursive enumeration we described above, except that we utilize $\mathbf{M}^{(k)}_{s-1..s} = \mathbf{M}_s$ to get some cancellation which simplifies the recursion.

Lemma 8.26. *For every (ℓ, r) s.t. $r > \ell$ and every $h, k \in \mathbb{N}$ s.t. $h \leq k$, we have the following identity:*

$$\mathbf{M}^{(k)}_{\ell..r} = \sum_{s=\ell+1}^{r} \mathbf{M}^{(h-1)}_{\ell..s-1}\mathbf{M}_s\mathbf{M}^{(k-h)}_{s..r} - \sum_{s=\ell+1}^{r-1} \mathbf{M}^{(h-1)}_{\ell..s}\mathbf{M}^{(k-h)}_{s..r}. \tag{8.2}$$

Surprisingly, this new recursion coincides with the recursion of Richardson iteration from the inverse Laplacian perspective. This shows that the construction in [CHL+23] is actually equivalent to the [CL20] construction that we use in this paper. We briefly discuss this equivalence in Section 8.5.

Before we prove the identity, we show that the new recursion does imply an algorithm that runs in space $\tilde{O}(\log(k)\log(nw))$. Consider the algorithm which recursively computes $\mathbf{M}^{(k)}_{\ell..r}$ using the formula in Lemma 8.26 with $h = \lceil k/2 \rceil$. Observe that the right hand side of Equation (8.2) involves at most $O(n)$ matrices. In addition, given all the matrices on the right hand side, the computation of $\mathbf{M}^{(k)}_{\ell..r}$ takes only $O(\log(nkwT))$ additional bits, where T is the maximum bit length of all the matrix entries. From Lemma 8.17 and Lemma 8.21 we can see that T is at most $\tilde{O}(k\log^2(n))$, so $O(\log(nwkT)) = \tilde{O}(\log(nwk))$. Finally, observe that each matrix on

[36] If there are two consecutive multiples of 2^t in (ℓ, r), then 2^{t+1} divides one of them, which violates the definition of t.

[37] There is an empty summation term $\sum_{i+j=-1}$ which should be treated as 0.

the right hand side has precision parameter at most $\max(h-1, k-h) \leq \lfloor k/2 \rfloor$. Therefore the recursion reaches the $\mathbf{M}^{(0)}_{\ell..r}$ base cases after at most $\lceil \log(k) \rceil$ levels. Therefore the space complexity is at most $\tilde{O}(\log(k)\log(nw)) + O(s_{\text{base}})$, where s_{base} is the maximum space complexity of computing $\mathbf{M}^{(0)}_{\ell..r}$. The base case complexity s_{base} is indeed $\tilde{O}(\log(nw))$, which we prove in Section 8.4.2.

Finally, we prove the identity in Lemma 8.26.

Proof of Lemma 8.26. We prove the claim by induction on $r - \ell$. Let $m = \mathsf{LCA}(\ell, r)$. For the base case $r - \ell = 1$, the lemma says $\mathbf{M}^{(k)}_{\ell..r} = \mathbf{M}^{(h-1)}_{\ell..\ell}\mathbf{M}_r\mathbf{M}_{r..r}$, which is trivially true. Next we prove the general case by induction. For each matrix $\mathbf{M}^{(k')}_{\ell'..r'}$ on the right hand side s.t. $\ell' < m < r'$, we expand $\mathbf{M}^{(k')}_{\ell'..r'}$ using the recursion in Claim 8.4. Note that $\mathsf{LCA}(\ell', r')$ is also m. In addition, for the $s = m$ term in the first summation we apply the dummy expansion $\mathbf{M}^{(k-h)}_{m..r} = \sum_{a+b=k-h} \mathbf{M}^{(a)}_{m..m}\mathbf{M}^{(b)}_{m..r} - \sum_{a+b=k-h-1} \mathbf{M}^{(a)}_{m..m}\mathbf{M}^{(b)}_{m..r}$. Similarly, for the $s = m+1$ term in the first summation we also expand $\mathbf{M}^{(k-h)}_{\ell..m} = \sum_{a+b=h-1} \mathbf{M}^{(a)}_{\ell..m}\mathbf{M}^{(b)}_{m..m} - \sum_{a+b=h-2} \mathbf{M}^{(a)}_{\ell..m}\mathbf{M}^{(b)}_{m..m}$. After rearranging we get that the right hand side equals to

$$
\begin{aligned}
&\sum_{s=\ell+1}^{m} \sum_{a+b=k-h} \mathbf{M}^{(h-1)}_{\ell..s-1}\mathbf{M}_s\mathbf{M}^{(a)}_{s..m}\mathbf{M}^{(b)}_{m..r} && - \sum_{s=\ell+1}^{m-1} \sum_{a+b=k-h} \mathbf{M}^{(h-1)}_{\ell..s}\mathbf{M}^{(a)}_{s..m}\mathbf{M}^{(b)}_{m..r} \\
&- \sum_{s=\ell+1}^{m} \sum_{a+b=k-h-1} \mathbf{M}^{(h-1)}_{\ell..s-1}\mathbf{M}_s\mathbf{M}^{(a)}_{s..m}\mathbf{M}^{(b)}_{m..r} && + \sum_{s=\ell+1}^{m-1} \sum_{a+b=k-h-1} \mathbf{M}^{(h-1)}_{\ell..s}\mathbf{M}^{(a)}_{s..m}\mathbf{M}^{(b)}_{m..r} \\
&+ \sum_{s=m+1}^{r} \sum_{a+b=h-1} \mathbf{M}^{(a)}_{\ell..m}\mathbf{M}^{(b)}_{m..s-1}\mathbf{M}_s\mathbf{M}^{(k-h)}_{s..r} && - \sum_{s=m+1}^{r} \sum_{a+b=h-1} \mathbf{M}^{(a)}_{\ell..m}\mathbf{M}^{(b)}_{m..s}\mathbf{M}^{(k-h)}_{s..r} \\
&- \sum_{s=m+1}^{r} \sum_{a+b=h-2} \mathbf{M}^{(a)}_{\ell..m}\mathbf{M}^{(b)}_{m..s-1}\mathbf{M}_s\mathbf{M}^{(k-h)}_{s..r} && + \sum_{s=m+1}^{r} \sum_{a+b=h-2} \mathbf{M}^{(a)}_{\ell..m}\mathbf{M}^{(b)}_{m..s}\mathbf{M}^{(k-h)}_{s..r} \\
& && - \mathbf{M}^{(h-1)}_{\ell..m}\mathbf{M}^{(k-h)}_{m..r}.
\end{aligned}
$$

Note that the first summation in Equation (8.2) expands to the terms on the left, and the second summation in Equation (8.2) expands to the terms on the right.

Now we classify all the terms in the first line by b, and take out the right factor $\mathbf{M}^{(b)}_{m..r}$. For any fixed $b \leq k-h$ we get the sum

$$\left(\sum_{s=\ell+1}^{m} \mathbf{M}^{(h-1)}_{\ell..s-1} \mathbf{M}_s \mathbf{M}^{(k-b-h)}_{s..m} - \sum_{s=\ell+1}^{m-1} \mathbf{M}^{(h-1)}_{\ell..s} \mathbf{M}^{(k-b-h)}_{s..m} \right) \mathbf{M}^{(b)}_{m..r}.$$

Observe that this is exactly $\mathbf{M}^{(k-b)}_{\ell..m} \mathbf{M}^{(b)}_{m..r}$ by induction. Therefore, we can see that the first line is exactly $\sum_{b=0}^{k-h} \mathbf{M}^{(k-b)}_{\ell..m} \mathbf{M}^{(b)}_{m..r}$. Similarly, we can get that the second line is $-\sum_{b=0}^{k-h-1} \mathbf{M}^{(k-b-1)}_{\ell..m} \mathbf{M}^{(b)}_{m..r}$. For the third and fourth lines, we can also classify the terms by a and take out the left factor $\mathbf{M}^{(a)}_{\ell..m}$ to get to get $\sum_{a=0}^{h-1} \mathbf{M}^{(a)}_{\ell..m} \mathbf{M}^{(k-a)}_{m..r}$ and $-\sum_{a=0}^{h-2} \mathbf{M}^{(a)}_{\ell..m} \mathbf{M}^{(k-a-1)}_{m..r}$ respectively. Finally, collect all the simplified terms (including the only term in the fifth line in the expansion), and we get

$$\begin{aligned}
&\sum_{b=0}^{k-h} \mathbf{M}^{(k-b)}_{\ell..m} \mathbf{M}^{(b)}_{m..r} + \sum_{a=0}^{h-1} \mathbf{M}^{(a)}_{\ell..m} \mathbf{M}^{(k-a)}_{m..r} \\
&- \sum_{b=0}^{k-h-1} \mathbf{M}^{(k-b)}_{\ell..m} \mathbf{M}^{(b)}_{m..r} - \sum_{a=0}^{h-2} \mathbf{M}^{(a)}_{\ell..m} \mathbf{M}^{(k-a-1)}_{m..r} - \mathbf{M}^{(h-1)}_{\ell..m} \mathbf{M}^{(k-h)}_{m..r} \\
&= \sum_{a+b=k} \mathbf{M}^{(a)}_{\ell..m} \mathbf{M}^{(b)}_{m..r} - \sum_{a+b=k-1} \mathbf{M}^{(a)}_{\ell..m} \mathbf{M}^{(b)}_{m..r} \\
&= \mathbf{M}^{(k)}_{\ell..r}.
\end{aligned}$$

□

8.4.2 Base-case Space Complexity

In this section we prove the following claim.

Lemma 8.27. *There is an algorithm which for every $\ell < r$ can compute $\mathbf{M}^{(0)}_{\ell..r}$ in space $\tilde{O}(\log(nw))$.*

Proof. We claim that it is possible to output the indices $(\ell_0, r_0) \in \mathsf{BS}_n$ of all the factors $\mathbf{M}^{(0)}_{\ell_0..r_0}$ in the expansion of $\mathbf{M}^{(0)}_{\ell..r}$ in space $O(\log(n))$, and there are only $2\log(n)$

factors. The claim then follows by Lemma 8.21, Lemma 2.48 and Lemma 2.51.

For the base cases $(\ell, r) \in \mathsf{BS}_n$ the claim is straightforward. For $(\ell, r) \notin \mathsf{BS}_n$, first compute $m = \mathsf{LCA}(\ell, r)$, and note that $\mathbf{M}^{(0)}_{\ell..r} = \mathbf{M}^{(0)}_{\ell..m}\mathbf{M}^{(0)}_{m..r}$. For the factors of $\mathbf{M}^{(0)}_{\ell..m}$, while $(\ell, m) \notin \mathsf{BS}_n$, we repeatedly compute $m' = \mathsf{LCA}(\ell, m)$, then output (m', m), and replace m with the value of m'. Eventually when $(\ell, m) \in \mathsf{BS}_n$ we output (ℓ, m).

Note that $\mathbf{M}^{(0)}_{\ell..m} = \mathbf{M}^{(0)}_{\ell..m'}\mathbf{M}^{(0)}_{m'..m}$, so if we can prove that $(m', m) \in \mathsf{BS}_n$, then what we output is exactly the indices of the expansion (except that the output order is reversed, which is easy to deal with). To prove that $(m', m) \in \mathsf{BS}_n$, assume that $m' = c' \cdot 2^{t'}$ and $m = c \cdot 2^t$, for some odd integers c', c. In our algorithm, we always have $m' = \mathsf{LCA}(\ell, m)$ and $m = \mathsf{LCA}(\ell, r')$ for some $r' > m$. Because $\ell < m' < m < r'$, by definition of LCA we must have $t > t'$. We claim that we must have $m = m' + 2^{t'}$, which implies $(m', m) \in \mathsf{BS}_n$.

If this is not the case, then we must have $m' < m' + 2^{t'} < m$ because m is also a multiple of m. Then observe that $m' + 2^{t'}$ is a multiple of $2^{t'+1}$, which contradicts to the fact that $m' = \mathsf{LCA}(\ell, m)$.

Now we have proved how to output the expansion indices of $\mathbf{M}^{(0)}_{\ell.m}$, and for $\mathbf{M}^{(0)}_{m..r}$ the proof is basically the same. Finally, to prove that there are at most $2\log(n)$ factors, observe that every time we replace m with m', the exponent t strictly decreases. Because we always have $0 \leq t < \log(n)$, the procedure above can repeat at most $\log(n)$ iterations. Therefore both $\mathbf{M}^{(0)}_{\ell.m}$ and $\mathbf{M}^{(0)}_{m..r}$ have at most $\log(n)$ factors.

□

8.5 Equivalence with the CHLTW construction

The construction in [CHL+23] is based on the inverse Laplacian perspective of derandomization [AKM+20] and preconditioned Richardson iteration, which we briefly recap as follows. Consider the block matrix $\mathbf{W} = (\mathbb{R}^{w\times w})^{(n+1)\times(n+1)}$ s.t. $\mathbf{W}[i-1,i] = \mathbf{M}_i$ for every $i \in [n]$, and other entries of $\mathbf{W}$ are 0. The Laplacian is defined as $\mathbf{L} := \mathbf{I} - \mathbf{W}$, which is an invertible matrix. It can be shown that each entry $\mathbf{L}^{-1}[i,j]$ in the inverse Laplacian $\mathbf{L}^{-1}$ is exactly $\mathbf{M}_{i..j}$ (where $\mathbf{M}_{i..i} = \mathbf{I}_w$ and $\mathbf{M}_{i..j} = 0$ if $i > j$).

To approximate $\mathbf{M}_{0..n}$ within error ε, [CHL+23] followed the preconditioned Richardson iteration approach, which first constructs a "mild approximation" of $\mathbf{L}^{-1}$ denoted by $\widetilde{\mathbf{L}^{-1}}$. Their construction of $\widetilde{\mathbf{L}^{-1}}$ is based on the shortcut graph structure, and one can verify that their $\widetilde{\mathbf{L}^{-1}}[i,j]$ is actually the same as $\mathbf{M}^{(0)}_{i..j}$ defined in this paper.

Given $\mathbf{L}^{-1}$, the construction based on Richardson iteration is defined as $(\mathbf{L}^{-1})^{(k)} = \widetilde{\mathbf{L}^{-1}}\sum_{i=0}^{k}(\mathbf{I} - \mathbf{L}\widetilde{\mathbf{L}^{-1}})^i$, and $(\mathbf{L}^{-1})^{(k)}[0,n]$ is the final output. Observe that this formula satisfies the recursion $(\mathbf{L}^{-1})^{(k)} = \widetilde{\mathbf{L}^{-1}} + (\mathbf{L}^{-1})^{(k-1)}(\mathbf{I} - \mathbf{L}\widetilde{\mathbf{L}^{-1}})$. In addition, observe that for every $i < j$, $(\mathbf{I} - \mathbf{L}\widetilde{\mathbf{L}^{-1}})[i,j] = \mathbf{M}_{i+1}\mathbf{M}^{(0)}_{i+1..j} - \mathbf{M}^{(0)}_{i..j}$,[38] and other entries of $\mathbf{I} - \mathbf{L}\widetilde{\mathbf{L}^{-1}}$ are 0. Therefore, for every $\ell \le r$ we have

$$(\mathbf{L}^{-1})^{(k)}[\ell,r] = \mathbf{M}^{(0)}_{\ell..r} + \sum_{s=\ell+1}^{r}(\mathbf{L}^{-1})^{(k-1)}[\ell,s-1]\left(\mathbf{M}_s\mathbf{M}^{(0)}_{s..r} - \mathbf{M}^{(0)}_{s-1..r}\right).$$

Comparing it with the special case $h = k$ of Lemma 8.26:

$$\mathbf{M}^{(k)}_{\ell..r} = \sum_{s=\ell+1}^{r}\mathbf{M}^{(k-1)}_{\ell..s-1}\mathbf{M}_s\mathbf{M}^{(0)}_{s..r} - \sum_{s=\ell+1}^{r-1}\mathbf{M}^{(k-1)}_{\ell..s}\mathbf{M}^{(0)}_{s..r}.$$

Using the fact that $(\mathbf{L}^{-1})^{(k-1)}[\ell,\ell] = \mathbf{I}$, it is relatively easy to prove via induction that $(\mathbf{L}^{-1})^{(k)}[\ell,r] = \mathbf{M}^{(k)}_{\ell,r}$.

[38] This is in fact the *local consistency error* defined in [CH20], as observed in [Hoz21a].

BIBLIOGRAPHY

[AB09] Sanjeev Arora and Boaz Barak. *Computational complexity: a modern approach*. Cambridge University Press, 2009.

[AGHP92] Noga Alon, Oded Goldreich, Johan Håstad, and René Peralta. Simple constructions of almost k-wise independent random variables. *Random Structures & Algorithms*, 3(3):289–304, 1992.

[AGM03] Noga Alon, Oded Goldreich, and Yishay Mansour. Almost k-wise independence versus k-wise independence. *Inf. Process. Lett.*, 88(3):107–110, 2003.

[AKM+20] AmirMahdi Ahmadinejad, Jonathan Kelner, Jack Murtagh, John Peebles, Aaron Sidford, and Salil Vadhan. High-precision estimation of random walks in small space. In *FOCS 2020*, 2020.

[AKS87] Miklós Ajtai, János Komlós, and Endre Szemerédi. Deterministic simulation in LOGSPACE. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York, USA*, pages 132–140. ACM, 1987.

[Ald83] David Aldous. Random walks on finite groups and rapidly mixing markov chains. In *Séminaire de Probabilités XVII 1981/82*, pages 243–297. Springer, 1983.

[APP+23] AmirMahdi Ahmadinejad, John Peebles, Edward Pyne, Aaron Sidford, and Salil Vadhan. Singular value approximation and reducing directed to undirected graph sparsification. In *FOCS 2023, to appear*, 2023.

[Arm98] Roy Armoni. On the derandomization of space-bounded computations. In *RANDOM 1998*, pages 47–59, 1998.

[BCG20] Mark Braverman, Gil Cohen, and Sumegha Garg. Pseudorandom pseudo-distributions with near-optimal error for read-once branching programs. *SIAM Journal on Computing*, 49(5):STOC18–242–STOC18–299, 2020.

[BCH86] Paul Beame, Stephen A. Cook, and H. James Hoover. Log depth circuits for division and related problems. *SIAM J. Comput.*, 15(4):994–1003, 1986.

[BDL16] Jean Bourgain, Zeev Dvir, and Ethan Leeman. Affine extractors over large fields with exponential error. *Comput. Complex.*, 25(4):921–931, 2016.

[BDT19] Avraham Ben-Aroya, Dean Doron, and Amnon Ta-Shma. An efficient reduction from two-source to nonmalleable extractors: achieving near-logarithmic min-entropy. *SIAM Journal on Computing*, pages STOC17–31, 2019.

[BDVY13] Andrej Bogdanov, Zeev Dvir, Elad Verbin, and Amir Yehudayoff. Pseudorandomness for width-2 branching programs. *Theory of Computing*, 9(1):283–293, 2013.

[BGDM23] Marshall Ball, Eli Goldin, Dana Dachman-Soled, and Saachi Mutreja. Extracting randomness from samplable distributions, revisited. In *64th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2023, Santa Cruz, CA, USA, November 6-9, 2023*, pages 1505–1514. IEEE, 2023.

[BHPP22] Andrej Bogdanov, William M Hoza, Gautam Prakriya, and Edward Pyne. Hitting sets for regular branching programs. In *37th Computational Complexity Conference (CCC 2022)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2022.

[Bou07] Jean Bourgain. On the construction of affine extractors. *GAFA Geometric And Functional Analysis*, 17(1):33–57, 2007.

[BPW11] Andrej Bogdanov, Periklis A. Papakonstantinou, and Andrew Wan. Pseudorandomness for read-once formulas. In *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011*, pages 240–246, 2011.

[BRRY14] Mark Braverman, Anup Rao, Ran Raz, and Amir Yehudayoff. Pseudorandom generators for regular branching programs. *SIAM Journal on Computing*, 43(3):973–986, 2014.

[BRTW14] Eli Ben-Sasson, Noga Ron-Zewi, Madhur Tulsiani, and Julia Wolf. Sampling-based proofs of almost-periodicity results and algorithmic applications. In *Automata, Languages, and Programming - 41st International Colloquium, ICALP 2014*, volume 8572 of *Lecture Notes in Computer Science*, pages 955–966. Springer, 2014.

[CDL01] Andrew Chiu, George I. Davida, and Bruce E. Litow. Division in logspace-uniform NC^1. *RAIRO Theor. Informatics Appl.*, 35(3):259–275, 2001.

[CDR+21] Gil Cohen, Dean Doron, Oren Renard, Ori Sberlo, and Amnon Ta-Shma. Error reduction for weighted prgs against read once branching programs. In *36th Computational Complexity Conference (CCC 2021)*, pages 22:1–22:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

[CG88] Benny Chor and Oded Goldreich. Unbiased bits from sources of weak randomness and probabilistic communication complexity. *SIAM J. Comput.*, 17(2):230–261, 1988.

[CG14] Mahdi Cheraghchi and Venkatesan Guruswami. Non-malleable coding against bit-wise and split-state tampering. In *Theory of Cryptography Conference*, pages 440–464. Springer, 2014.

[CG21] Eshan Chattopadhyay and Jesse Goodman. Improved extractors for small-space sources. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021*, pages 610–621. IEEE, 2021.

[CGH+85] Benny Chor, Oded Goldreich, Johan Håstad, Joel Friedman, Steven Rudich, and Roman Smolensky. The bit extraction problem of t-resilient functions (preliminary version). In *26th Annual Symposium on Foundations of Computer Science, STOC 1985*, pages 396–407. IEEE Computer Society, 1985.

[CGL20] Eshan Chattopadhyay, Vipul Goyal, and Xin Li. Nonmalleable extractors and codes, with their many tampered extensions. *SIAM J. Comput.*, 49(5):999–1040, 2020.

[CGL21] Eshan Chattopadhyay, Jesse Goodman, and Jyun-Jie Liao. Affine extractors for almost logarithmic entropy. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021*, pages 622–633. IEEE, 2021.

[CGZ22] Eshan Chattopadhyay, Jesse Goodman, and David Zuckerman. The space complexity of sampling. In *13th Innovations in Theoretical Computer Science Conference (ITCS 2022)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2022.

[CH20] Kuan Cheng and William M Hoza. Hitting sets give two-sided derandomization of small space. In *35th Computational Complexity Conference (CCC 2020)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2020.

[Cha02] Mei-Chu Chang. A polynomial bound in Freiman's theorem. *Duke mathematical journal*, 113(3):399–419, 2002.

[Che10] Mahdi Cheraghchi. *Applications of Derandomization Theory in Coding*. PhD thesis, EPFL, Lausanne, Switzerland, 2010.

[CHL+23] Lijie Chen, William M. Hoza, Xin Lyu, Avishay Tal, and Hongxun Wu. Weighted pseudorandom generators via inverse analysis of random walks and shortcutting. In *64th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2023*, pages 1224–1239, 2023.

[CHRT18] Eshan Chattopadhyay, Pooya Hatami, Omer Reingold, and Avishay Tal. Improved pseudorandomness for unordered branching programs through local monotonicity. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 363–375, 2018.

[CL16a] Eshan Chattopadhyay and Xin Li. Explicit non-malleable extractors, multi-source extractors, and almost optimal privacy amplification protocols. In *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016*, pages 158–167. IEEE Computer Society, 2016.

[CL16b] Eshan Chattopadhyay and Xin Li. Extractors for sumset sources. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016*, pages 299–311. ACM, 2016.

[CL17] Eshan Chattopadhyay and Xin Li. Non-malleable codes and extractors for small-depth circuits, and affine functions. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1171–1184, 2017.

[CL20] Eshan Chattopadhyay and Jyun-Jie Liao. Optimal error pseudodistribu-

tions for read-once branching programs. In *35th Computational Complexity Conference, CCC 2020*, volume 169 of *LIPIcs*, pages 25:1–25:27. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

[CL22] Eshan Chattopadhyay and Jyun-Jie Liao. Extractors for sum of two sources. In *STOC '22: 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1584–1597. ACM, 2022.

[CL23] Eshan Chattopadhyay and Jyun-Jie Liao. Hardness against linear branching programs and more. In *Leibniz International Proceedings in Informatics (LIPIcs): 38th Computational Complexity Conference (CCC 2023)*. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2023.

[CL24] Eshan Chattopadhyay and Jyun-Jie Liao. Recursive error reduction for regular branching programs. In *15th Innovations in Theoretical Computer Science Conference, ITCS 2024*, volume 287 of *LIPIcs*, pages 29:1–29:20, 2024.

[Coh16a] Gil Cohen. Local correlation breakers and applications to three-source extractors and mergers. *SIAM J. Comput.*, 45(4):1297–1338, 2016.

[Coh16b] Gil Cohen. Making the most of advice: New correlation breakers and their applications. In *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016*, pages 188–196. IEEE Computer Society, 2016.

[Coh17] Gil Cohen. Towards optimal two-source extractors and Ramsey graphs. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017*, pages 1157–1170. ACM, 2017.

[CS10] Ernie Croot and Olof Sisask. A probabilistic technique for find-

ing almost-periods of convolutions. *Geometric and functional analysis*, 20(6):1367–1396, 2010.

[CS16a] Gil Cohen and Leonard J. Schulman. Extractors for near logarithmic min-entropy. In *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016*, pages 178–187. IEEE Computer Society, 2016.

[CS16b] Gil Cohen and Igor Shinkar. The complexity of DNF of parities. In *Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science, Cambridge, MA, USA, January 14-16, 2016*, pages 47–58. ACM, 2016.

[CZ16] Eshan Chattopadhyay and David Zuckerman. New extractors for interleaved sources. In *31st Conference on Computational Complexity, CCC 2016*, volume 50 of *LIPIcs*, pages 7:1–7:28. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016.

[CZ19] Eshan Chattopadhyay and David Zuckerman. Explicit two-source extractors and resilient functions. *Annals of Mathematics*, 189(3):653–705, 2019.

[DG10] Matt DeVos and Ariel Gabizon. Simple affine extractors using dimension expansion. In *2010 IEEE 25th Annual Conference on Computational Complexity*, pages 50–57. IEEE, 2010.

[DGJ+10] Ilias Diakonikolas, Parikshit Gopalan, Ragesh Jaiswal, Rocco A. Servedio, and Emanuele Viola. Bounded independence fools halfspaces. *SIAM J. Comput.*, 39(8):3441–3462, 2010.

[DK11] Evgeny Demenkov and Alexander S. Kulikov. An elementary proof of a 3n - o(n) lower bound on the circuit complexity of affine dispersers.

In *Mathematical Foundations of Computer Science 2011 - 36th International Symposium, MFCS 2011, Warsaw, Poland, August 22-26, 2011. Proceedings*, volume 6907 of *Lecture Notes in Computer Science*, pages 256–265. Springer, 2011.

[DKSS13] Zeev Dvir, Swastik Kopparty, Shubhangi Saraf, and Madhu Sudan. Extensions to the method of multiplicities, with applications to kakeya sets and mergers. *SIAM J. Comput.*, 42(6):2305–2328, 2013.

[DLWZ11] Yevgeniy Dodis, Xin Li, Trevor D. Wooley, and David Zuckerman. Privacy amplification and non-malleable extractors via character sums. In *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25, 2011*, pages 668–677. IEEE Computer Society, 2011.

[DORS08] Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam D. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM J. Comput.*, 38(1):97–139, 2008.

[DP07] Stefan Dziembowski and Krzysztof Pietrzak. Intrusion-resilient secret sharing. In *48th Annual IEEE Symposium on Foundations of Computer Science FOCS 2007*, pages 227–237. IEEE Computer Society, 2007.

[DW09] Yevgeniy Dodis and Daniel Wichs. Non-malleable extractors and symmetric key cryptography from weak secrets. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009*, pages 601–610. ACM, 2009.

[EOT10] Jordan S Ellenberg, Richard Oberlin, and Terence Tao. The kakeya set and maximal conjectures for algebraic varieties over finite fields. *Mathematika*, 56(1):1–25, 2010.

[FGHK16] Magnus Gausdal Find, Alexander Golovnev, Edward A. Hirsch, and Alexander S. Kulikov. A better-than-3n lower bound for the circuit complexity of an explicit function. In *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, pages 89–98. IEEE Computer Society, 2016.

[FK18] Michael A. Forbes and Zander Kelley. Pseudorandom generators for read-once branching programs, in any order. In *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 946–955. IEEE Computer Society, 2018.

[Fre73] Gregory A Freiman. Foundations of a structual theory of set addition. *Translation of Math. Monographs*, 37, 1973.

[GGMT23] WT Gowers, Ben Green, Freddie Manners, and Terence Tao. On a conjecture of marton. *arXiv preprint arXiv:2311.05762*, 2023.

[Gol11] Oded Goldreich. A sample of samplers: A computational perspective on sampling. In Oded Goldreich, editor, *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation - In Collaboration with Lidor Avigad, Mihir Bellare, Zvika Brakerski, Shafi Goldwasser, Shai Halevi, Tali Kaufman, Leonid Levin, Noam Nisan, Dana Ron, Madhu Sudan, Luca Trevisan, Salil Vadhan, Avi Wigderson, David Zuckerman*, volume 6650 of *Lecture Notes in Computer Science*, pages 302–332. Springer, 2011.

[GPT22] Svyatoslav Gryaznov, Pavel Pudlák, and Navid Talebanfard. Linear branching programs and directional affine extractors. In *37th Computational Complexity Conference, CCC 2022, July 20-23, 2022, Philadelphia, PA,*

USA, volume 234 of *LIPIcs*, pages 4:1–4:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

[GR07] Ben Green and Imre Z Ruzsa. Freiman's theorem in an arbitrary abelian group. *Journal of the London Mathematical Society*, 75(1):163–175, 2007.

[GRS06] Ariel Gabizon, Ran Raz, and Ronen Shaltiel. Deterministic extractors for bit-fixing sources by obtaining an independent seed. *SIAM J. Comput.*, 36(4):1072–1094, 2006.

[GTW21] Uma Girish, Avishay Tal, and Kewen Wu. Fourier growth of parity decision trees. *arXiv preprint arXiv:2103.11604*, 2021.

[GUV09] Venkatesan Guruswami, Christopher Umans, and Salil P. Vadhan. Unbalanced expanders and randomness extractors from Parvaresh-Vardy codes. *J. ACM*, 56(4):20:1–20:34, 2009.

[GW97] Oded Goldreich and Avi Wigderson. Tiny families of functions with random properties: A quality-size trade-off for hashing. *Random Struct. Algorithms*, 11(4):315–343, 1997.

[Has89] John Hastad. Almost optimal lower bounds for small depth circuits. *Adv. Comput. Res.*, 5:143–170, 1989.

[HHL18] Hamed Hatami, Kaave Hosseini, and Shachar Lovett. Structure of protocols for xor functions. *SIAM Journal on Computing*, 47(1):208–217, 2018.

[Hoz21a] William M Hoza. Better pseudodistributions and derandomization for space-bounded computation. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2021)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2021.

[Hoz21b] William Michael Hoza. *Derandomizing space-bounded computation via pseudorandom generators and their generalizations*. PhD thesis, 2021.

[HU17] William M. Hoza and Chris Umans. Targeted pseudorandom generators, simulation advice generators, and derandomizing logspace. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017*, pages 629–640, 2017.

[HZ20] William M. Hoza and David Zuckerman. Simple optimal hitting sets for small-success RL. *SIAM J. Comput.*, 49(4):811–820, 2020.

[ILL89] Russell Impagliazzo, Leonid A. Levin, and Michael Luby. Pseudorandom generation from one-way functions (extended abstracts). In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing, May 14-17, 1989, Seattle, Washington, USA*, pages 12–24. ACM, 1989.

[INW94] Russell Impagliazzo, Noam Nisan, and Avi Wigderson. Pseudorandomness for network algorithms. In *Proceedings of the twenty-sixth annual ACM symposium on Theory of computing*, pages 356–364, 1994.

[Juk12] Stasys Jukna. *Boolean Function Complexity - Advances and Frontiers*, volume 27 of *Algorithms and combinatorics*. Springer, 2012.

[Kar71] Anatolii Alekseevich Karatsuba. On a certain arithmetic sum. In *Doklady Akademii Nauk*, volume 199, pages 770–772. Russian Academy of Sciences, 1971.

[Kar91] Anatolii Alekseevich Karatsuba. Distribution of values of Dirichlet characters on additive sequences. In *Doklady Akademii Nauk*, volume 319, pages 543–545. Russian Academy of Sciences, 1991.

[KLSS11] Swastik Kopparty, Vsevolod F Lev, Shubhangi Saraf, and Madhu Sudan. Kakeya-type sets in finite vector spaces. *Journal of Algebraic Combinatorics*, 34(3):337–355, 2011.

[KM93] Eyal Kushilevitz and Yishay Mansour. Learning decision trees using the fourier spectrum. *SIAM J. Comput.*, 22(6):1331–1348, 1993.

[KNW08] Daniel M. Kane, Jelani Nelson, and David P. Woodruff. Revisiting norm estimation in data streams. *CoRR*, abs/0811.3648, 2008.

[KRVZ11] Jesse Kamp, Anup Rao, Salil P. Vadhan, and David Zuckerman. Deterministic extractors for small-space sources. *J. Comput. Syst. Sci.*, 77(1):191–220, 2011.

[KZ07] Jesse Kamp and David Zuckerman. Deterministic extractors for bit-fixing sources and exposure-resilient cryptography. *SIAM J. Comput.*, 36(5):1231–1247, 2007.

[Li11] Xin Li. A new approach to affine extractors and dispersers. In *Proceedings of the 26th Annual IEEE Conference on Computational Complexity, CCC 2011*, pages 137–147. IEEE Computer Society, 2011.

[Li13] Xin Li. Extractors for a constant number of independent sources with polylogarithmic min-entropy. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013*, pages 100–109. IEEE Computer Society, 2013.

[Li15] Xin Li. Non-malleable condensers for arbitrary min-entropy, and almost optimal protocols for privacy amplification. In *Theory of Cryptography - 12th Theory of Cryptography Conference, TCC 2015*, volume 9014 of *Lecture Notes in Computer Science*, pages 502–531. Springer, 2015.

[Li16] Xin Li. Improved two-source extractors, and affine extractors for polylogarithmic entropy. In *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016*, pages 168–177. IEEE Computer Society, 2016.

[Li17] Xin Li. Improved non-malleable extractors, non-malleable codes and independent source extractors. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017*, pages 1144–1156. ACM, 2017.

[Li19] Xin Li. Non-malleable extractors and non-malleable codes: Partially optimal constructions. In *34th Computational Complexity Conference, CCC 2019*, volume 137 of *LIPIcs*, pages 28:1–28:49. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

[Li23] Xin Li. Two source extractors for asymptotically optimal entropy, and (many) more. In *64th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2023*, pages 1271–1281, 2023.

[Lov15] Shachar Lovett. An exposition of Sanders' quasi-polynomial Freiman-Ruzsa theorem. *Theory Comput.*, 6:1–14, 2015.

[LPV23] Chin Ho Lee, Edward Pyne, and Salil Vadhan. On the power of regular and permutation branching programs. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2023)*, 2023.

[LRVW03] Chi-Jen Lu, Omer Reingold, Salil P. Vadhan, and Avi Wigderson. Extractors: optimal up to constant factors. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing, STOC 2003*, pages 602–611. ACM, 2003.

[LV17] Chin Ho Lee and Emanuele Viola. Some limitations of the sum of small-bias distributions. *Theory of Computing*, 13(1):1–23, 2017.

[LY22] Jiatu Li and Tianqi Yang. $3.1n$ - $o(n)$ circuit lower bounds for explicit functions. In *STOC '22: 54th Annual ACM SIGACT Symposium on Theory of Computing, Rome, Italy, June 20 - 24, 2022*, pages 1180–1193. ACM, 2022.

[LZ23] Xin Li and Yan Zhong. Explicit directional affine extractors and improved hardness for linear branching programs. *arXiv preprint arXiv:2304.11495*, 2023.

[Mek17] Raghu Meka. Explicit resilient functions matching Ajtai-Linial. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017*, pages 1132–1148. SIAM, 2017.

[MRSV17] Jack Murtagh, Omer Reingold, Aaron Sidford, and Salil Vadhan. Derandomization beyond connectivity: Undirected laplacian systems in nearly logarithmic space. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 801–812. IEEE, 2017.

[MRT19] Raghu Meka, Omer Reingold, and Avishay Tal. Pseudorandom generators for width-3 branching programs. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019*. ACM, 2019.

[MW97] Ueli M. Maurer and Stefan Wolf. Privacy amplification secure against active adversaries. In *Advances in Cryptology - CRYPTO '97, 17th Annual International Cryptology Conference*, volume 1294 of *Lecture Notes in Computer Science*, pages 307–321. Springer, 1997.

[Nis92] Noam Nisan. Pseudorandom generators for space-bounded computation. *Comb.*, 12(4):449–461, 1992.

[Nis94] Noam Nisan. RL $\subseteq$ SC. *Comput. Complex.*, 4:1–11, 1994.

[NN93] Joseph Naor and Moni Naor. Small-bias probability spaces: Efficient constructions and applications. *SIAM J. Comput.*, 22(4):838–856, 1993.

[NZ96] Noam Nisan and David Zuckerman. Randomness is linear in space. *J. Comput. Syst. Sci.*, 52(1):43–52, 1996.

[Plü61] Helmut Plünnecke. *Eigenschaften und Abschätzungen von Wirkungsfunktionen*. Number 22. Gesellschaft für Mathematik u. Datenverarbeitung, 1961.

[PV21] Edward Pyne and Salil Vadhan. Pseudodistributions that beat all pseudorandom generators. In *36th Computational Complexity Conference (CCC 2021)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2021.

[Rao09] Anup Rao. Extractors for low-weight affine sources. In *Proceedings of the 24th Annual IEEE Conference on Computational Complexity, CCC 2009*, pages 95–101. IEEE Computer Society, 2009.

[RSV13] Omer Reingold, Thomas Steinke, and Salil Vadhan. Pseudorandomness for regular branching programs via fourier analysis. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 655–670. Springer, 2013.

[RT00] Jaikumar Radhakrishnan and Amnon Ta-Shma. Bounds for dispersers, extractors, and depth-two superconcentrators. *SIAM J. Discret. Math.*, 13(1):2–24, 2000.

[RTV06] Omer Reingold, Luca Trevisan, and Salil Vadhan. Pseudorandom walks on regular digraphs and the RL vs. L problem. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, pages 457–466, 2006.

[Ruz99] Imre Ruzsa. An analog of Freiman's theorem in groups. *Astérisque*, 258(199):323–326, 1999.

[RV05] Eyal Rozenman and Salil Vadhan. Derandomized squaring of graphs. In *International Workshop on Approximation Algorithms for Combinatorial Optimization*, pages 436–447. Springer, 2005.

[RY11] Ran Raz and Amir Yehudayoff. Multilinear formulas, maximal-partition discrepancy and mixed-sources extractors. *J. Comput. Syst. Sci.*, 77(1):167–190, 2011.

[San12] Tom Sanders. On the Bogolyubov-Ruzsa lemma. *Analysis & PDE*, 5(3):627–655, 2012.

[Sav70] Walter J. Savitch. Relationships between nondeterministic and deterministic tape complexities. *J. Comput. Syst. Sci.*, 4(2):177–192, 1970.

[SZ99] Michael E. Saks and Shiyu Zhou. $\mathrm{BP}_H\mathrm{Space(S)} \subseteq \mathrm{DSPACE}(S^{3/2})$. *J. Comput. Syst. Sci.*, 58(2):376–403, 1999.

[Tre01] Luca Trevisan. Extractors and pseudorandom generators. *J. ACM*, 48(4):860–879, 2001.

[TS17] Amnon Ta-Shma. Explicit, almost optimal, epsilon-balanced codes. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 238–251, 2017.

[TV00] Luca Trevisan and Salil P. Vadhan. Extracting randomness from samplable distributions. In *41st Annual Symposium on Foundations of Computer Science, FOCS 2000*, pages 32–42. IEEE Computer Society, 2000.

[TV06] Terence Tao and Van H. Vu. *Additive combinatorics*, volume 105. Cambridge University Press, 2006.

[TWXZ13] Hing Yin Tsang, Chung Hoi Wong, Ning Xie, and Shengyu Zhang. Fourier sparsity, spectral norm, and the log-rank conjecture. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 658–667. IEEE, 2013.

[Tzu09] Yoav Tzur. Notions of weak pseudorandomness and gf (2n)-polynomials. *Master's thesis, Weizmann Institute of Science*, 2009.

[Vad12] Salil P. Vadhan. Pseudorandomness. *Found. Trends Theor. Comput. Sci.*, 7(1-3):1–336, 2012.

[Vaz87] Umesh V. Vazirani. Strong communication complexity or generating quasirandom sequences form two communicating semi-random sources. *Comb.*, 7(4):375–392, 1987.

[Vio14] Emanuele Viola. Extractors for circuit sources. *SIAM J. Comput.*, 43(2):655–672, 2014.

[vN28] John von Neumann. Zur theorie der gesellschaftsspiele. *Mathematische annalen*, 100(1):295–320, 1928.

[Yao85] Andrew Chi-Chih Yao. Separating the polynomial-time hierarchy by oracles (preliminary version). In *26th Annual Symposium on Foundations of Computer Science, Portland, Oregon, USA, 21-23 October 1985*, pages 1–10. IEEE Computer Society, 1985.

[Yeh11] Amir Yehudayoff. Affine extractors over prime fields. *Combinatorica*, 31(2):245–256, 2011.

[Zuc97] David Zuckerman. Randomness-optimal oblivious sampling. *Random Struct. Algorithms*, 11(4):345–367, 1997.

[Zuc07] David Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. *Theory Comput.*, 3(1):103–128, 2007.