Electronic Colloquium on Computational Complexity, Report No. 97 (2024)

Near-Optimal Averaging Samplers

Zhiyang Xun*
Department of Computer Science
University of Texas at Austin
zxun@cs.utexas.edu

David Zuckerman†
Department of Computer Science
University of Texas at Austin
diz@utexas.edu

May 31, 2024

Abstract

We present the first efficient averaging sampler that achieves asymptotically optimal randomness complexity and near-optimal sample complexity for natural parameter choices. Specifically, for any constant $\alpha > 0$, for $\delta > 2^{-\operatorname{poly}(1/\varepsilon)}$, it uses $m + O(\log(1/\delta))$ random bits to output $t = O(\log(1/\delta)/\varepsilon^{2+\alpha})$ samples $Z_1, \ldots Z_t \in \{0,1\}^m$ such that for any function $f : \{0,1\}^m \to [0,1]$,

$$\Pr\left[\left|\frac{1}{t}\sum_{i=1}^{t} f(Z_i) - \mathbb{E} f\right| \le \varepsilon\right] \ge 1 - \delta.$$

The sample complexity is optimal up to the $O(\varepsilon^{\alpha})$ factor.

We use known connections with randomness extractors and list-decodable codes to give applications to these objects.

1 Introduction

Randomization plays a crucial role in computer science, offering significant benefits across various applications. However, obtaining true randomness can be challenging. It's therefore natural to study whether we can achieve the benefits of randomization while using few random bits.

One of the most basic uses of randomness is sampling. Given oracle access to an arbitrary function $f : \{0,1\}^m \to [0,1]$ on a large domain, our goal is to estimate its average value. By drawing $t = O(\log(1/\delta)/\epsilon^2)$ independent random samples $Z_1, \ldots, Z_t \in \{0,1\}^m$, the Chernoff bound guarantees that the average value $\left|\frac{1}{t}\sum_{i=1}^{t} f(Z_i) - \mathbb{E} f\right| \le \varepsilon$ with probability at least $1 - \delta$. This method uses full independence in sampling, but we can also pursue more efficient strategies. This leads to the following definition:

Definition 1.1 ([BR94]). *A function* $\mathsf{Samp} : \{0,1\}^n \to (\{0,1\}^m)^t$ *is a* (δ, ε) *averaging sampler with* t *samples using* n *random bits if for every function* $f : \{0,1\}^m \to [0,1]$*, we have*

$$\Pr_{(Z_1,\ldots,Z_t)\sim\mathsf{Samp}(U_n)}\left[\left|\frac{1}{t}\sum_{i} f(Z_i) - \mathbb{E} f\right| \le \varepsilon\right] \ge 1 - \delta.$$

*Supported by NSF award CCF-2008868 and the NSF AI Institute for Foundations of Machine Learning (IFML).
†Supported by NSF Grant CCF-2312573 and a Simons Investigator Award (#409864).

ISSN 1433-8092

We would like to construct explicit samplers using few random bits that have sample complexity close to the optimal. Researchers have made progress towards this goal, and a summary is given in Table 1. Bellare and Rompel [BR94] suggested that interesting choices of parameters are $\varepsilon = 1/\text{poly}(m)$ and $\delta = \exp(-\text{poly}(m))$. This enables us to use $\text{poly}(m)$ random bits and generate $\text{poly}(m)$ samples.

Due to	Method	Random Bits	Sample Complexity
[CEG95]	Lower Bound	$m + \log(1/\delta) - \log(O(t))$	$\Omega(\log(1/\delta)/\varepsilon^2)$
[CEG95]	Non-Explicit	$m + 2\log(2/\delta) + \log\log(1/\varepsilon)$	$2\log(4/\delta)/\varepsilon^2$
Standard	Full Independence	$O(m\log(1/\delta)/\varepsilon^2)$	$O(\log(1/\delta)/\varepsilon^2)$
[CG89]	Pairwise Independence	$2m$	$O(1/(\delta\varepsilon^2))$
[Gil98]	Expander Walks	$m + O(\log(1/\delta)/\varepsilon^2)$	$O(\log(1/\delta)/\varepsilon^2)$
[BR94]	Iterated Sampling	$O(m + (\log m)\log(1/\delta))$	$\text{poly}(1/\varepsilon, \log(1/\delta), \log m)$
[Zuc97]	Hash-Based Extractors	$(1+\alpha)(m + \log(1/\delta))$	$\text{poly}(1/\varepsilon, \log(1/\delta), m)$
[RVW00]	Zig-Zag Extractors	$m + (1+\alpha)\log(1/\delta)$	$\text{poly}(1/\varepsilon, \log(1/\delta))$
Here	[RVW00] + Almost ℓ-wise Uniform	$m + O(\log(1/\delta))$	$O(\log(1/\delta)/\varepsilon^{2+\alpha})$

Table 1: Comparison of averaging samplers, α any positive constant, $\varepsilon = 1/\text{poly}(m)$, and $\delta = \exp(-\text{poly}(m))$.

The best existing randomness-efficient averaging sampler comes from the equivalence between averaging samplers and extractors [Zuc97]. Improving Zuckerman's construction, Reingold, Vadhan, and Wigderson [RVW00] gave a (δ, ε) averaging sampler for domain $\{0,1\}^m$ that uses $m + (1+\alpha)\log(1/\delta)$ random bits for any positive constant α. This almost matches the lower bound in [CEG95]. However, a notable gap remains in sample complexity: the existing construction's complexity $\text{poly}(1/\varepsilon, \log(1/\delta))$ does not align with the optimal $O(\log(1/\delta)/\varepsilon^2)$. This raised an open problem: Can we design an averaging sampler that not only meets the $O(m + \log(1/\delta))$ random bit requirement but also achieves the more efficient sample complexity of $O(\log(1/\delta)/\varepsilon^2)$ [BR94, Zuc97, Gol11]?

We note that such algorithms do exist for general samplers, which queries f and computes the estimation of $\mathbb{E} f$ by an arbitrary computation [BGG93]. However, many applications require the use of averaging samplers, such as the original use in interactive proofs [BR94]. Beyond these applications, averaging samplers act as a fundamental combinatorial object that relate to other notions such as randomness extractors, expander graphs, and list-decodable codes [Zuc97, Vad07].

1.1 Our Sampler

In this paper, we construct a polynomial-time computable (δ, ε) averaging sampler with near-optimal sample complexity using an asymptotically optimal number of random bits. In fact, the sampler we constructed is a *strong* sampler, defined as follows:

Definition 1.2. *A (δ, ε) averaging sampler* Samp *is* strong *if for every sequence of t functions $f_1, \ldots, f_t : \{0,1\}^m \to [0,1]$, we have*

$$\Pr_{(Z_1,\ldots,Z_t)\sim \mathsf{Samp}(U_n)}\left[\left|\frac{1}{t}\sum_i (f_i(Z_i) - \mathbb{E} f_i)\right| \le \varepsilon\right] \ge 1 - \delta.$$

We then state our main theorem:

Theorem 1. *For every constant $\alpha > 0$, there exists an efficient strong (δ, ε) averaging sampler for domain $\{0,1\}^m$ with $O(\frac{1}{\varepsilon^{2+\alpha}} \log \frac{1}{\delta})$ samples using $m + O((1 + \frac{\log\log(1/\delta)}{\log(1/\varepsilon)}) \log \frac{1}{\delta})$ random bits.*

We have the next immediate corollary.

Corollary 2. *For arbitrary positive constants α and C, given any (δ, ε) such that $\log(1/\delta) < \varepsilon^{-C}$, there exists an explicit (δ, ε) averaging sampler for domain $\{0,1\}^m$ with $O(\frac{1}{\varepsilon^{2+\alpha}} \log \frac{1}{\delta})$ samples using $m + O(\log \frac{1}{\delta})$ random bits.*

In particular, when $\varepsilon = 1/\mathrm{poly}(m)$ and $\delta = \exp(-\mathrm{poly}(m))$, our sampler achieves $O(\frac{1}{\varepsilon^{2+\alpha}} \log \frac{1}{\delta})$ sample complexity while using $m + O(\log \frac{1}{\delta})$ random bits, which is optimal up to the ε^{α} factor.

1.2 Randomness Extractors

Our sampler construction has implications for randomness extractors. A randomness extractor is a function that extracts almost-uniform bits from a low-quality source of randomness. We define the quality of a random source as its min-entropy.

Definition 1.3. *The min-entropy of a random variable X is*

$$H_\infty(X) := \min_{x \in \mathrm{supp}(X)} \log\left(\frac{1}{\Pr[X = x]}\right).$$

An (n, k)-source is a random variable on n bits with min-entropy at least k.

Then a randomness extractor is defined as:

Definition 1.4 ([NZ96])**.** *A function $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is a (k, ε) extractor if for every (n, k)-source X, the distribution $\mathsf{Ext}(X, U_d) \approx_\varepsilon U_m$. We say Ext is a* strong *(k, ε) extractor if for every (n, k)-source X, the distribution $(\mathsf{Ext}(X, Y), Y) \approx_\varepsilon U_{m+d}$, where Y is chosen from U_d.*

Randomness extractors are used in many areas within theoretical computer science. However, there has been little study of explicit extractors with the right dependence on ε. This is a particular concern in cryptography, where ε is often very small. Existentially, there are extractors with seed length $d = \log(n - k) + 2\log(1/\varepsilon) + O(1)$, and there is a matching lower bound [RT00].

Zuckerman [Zuc97] showed that averaging samplers are essentially equivalent to extractors. Specifically, an extractor $\mathsf{Ext} : \{0,1\}^n \times [2^d] \to \{0,1\}^m$ can be seen as a sampler that generates $\mathsf{Ext}(X, i)$ as its i-th sample point using the random source X. Using this equivalence, we can show that our sampler implies an extractor with almost optimal dependence on ε.

Theorem 3. *For any positive constants C and α, all $\varepsilon \geq 0$, and all k such that $k \geq n - \varepsilon^{-C}$, there exists an efficient strong (k, ε) extractor $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ with $m = \Omega(k)$ and $d = \log(n - k) + (2 + \alpha)\log(1/\varepsilon) + O(1)$.*

Prior to our work, extractors with a seed length dependence on ε achieving $2\log(1/\varepsilon)$ or close to it were based on the leftover hash lemma [BBR88, IZ89, HILL99] and expander random walks [Gil98, Zuc07]. Extractors using the leftover hash lemma have a seed length of $n + 2\log(1/\varepsilon)$, which is far from optimal. Expander random walks give a (k, ε) extractor with $k > (1 - \Omega(\varepsilon^2))n$ and an optimal seed length of $\log(n - k) + 2\log(1/\varepsilon) + O(1)$. Our extractor allows for smaller k whenever $\varepsilon < 1/n^c$ for an arbitrarily small constant $c > 0$.

1.3 Techniques

Our construction is very simple, and is based on two observations:

1. Rather than querying every sample point produced by a sampler Samp, we can use a second sampler Samp' to pick certain samples for querying. This reduces the sample complexity because the number of queried samples just depends on Samp'. Since the domain of Samp' is much smaller than the original domain, this allows more efficient sampling strategies. This observation has been utilized in previous sampler constructions [BR94, Gol11].

2. The bottleneck of generating an almost ℓ-wise uniform sequence over a large domain $\{0,1\}^m$ lies in sampling ℓ independent random points, which costs ℓm random bits. Since we can only afford $O(m)$ random bits, we are restricted to generating constant-wise uniform samples. However, for a much smaller domain, we can use few random bits to generate an almost ℓ-wise uniform sequence for large ℓ.

Our construction is outlined as follows. Let $\mathsf{Ext} : \{0,1\}^n \times [t'] \to \{0,1\}^m$ be the extractor-based sampler in [RVW00]. Let $Y_1, \ldots, Y_t$ be an almost ℓ-wise uniform sequence over domain $[t']$, thinking of $t \ll t'$. Our sampler is then defined by

$$\mathsf{Samp} := (\mathsf{Ext}(X, Y_1), \mathsf{Ext}(X, Y_2), \ldots, \mathsf{Ext}(X, Y_t)).$$

In this construction, we use the almost ℓ-wise uniform sequence to sub-sample from the extractor-based sampler. This can be viewed as a composition, similar to other cases such as Justesen codes [Jus72] and the first PCP theorem [ALM+98], where the goal is to optimize two main parameters simultaneously by combining two simpler schemes, each optimizing one parameter without significantly compromising the other.

Previous works have also applied almost ℓ-wise independence in extractor constructions. Srinivasan and Zuckerman [SZ99] proved a randomness-efficient leftover hash lemma by sampling an almost ℓ-wise independent function using uniform seeds and inputting a weak random source. Our construction inverts this process: we generate an ℓ-wise uniform sequence using a weak random source and then choose an index uniformly. Furthermore, Ran Raz's two-source extractor [Raz05] utilized two weak random sources to sample an almost ℓ-wise uniform sequence and an index separately. This is a more general construction, but if we directly apply Raz's error bound in our analysis Lemma 3.3, the final sample complexity will be off by a $\log(1/\delta)$ factor.

It might be of independent interest to readers who are familiar with the Nisan-Zuckerman pseudorandom generator [NZ96]. Our sampler has the same structure as the Nisan-Zuckerman generator, and one can view our construction from the perspective of pseudorandom generators. In the classical analysis of the Nisan-Zuckerman generator, ensuring a success probability of $1-\delta$ demands an extractor error smaller than δ, since an error at any step implies a complete loss of control. However, in our setting, *every* output sample has a very small effect on the final answer. This enables us to use an extractor with much larger error than δ here.

1.4 List-Decodable Codes

Another perspective on averaging samplers is its connection to error-correcting codes. Ta-Shma and Zuckerman [TZ04] showed that strong randomness extractors are equivalent to codes with good soft-decision decoding, which is related to list recovery. From this perspective, the composition scheme in our construction is similar to code concatenation.

For codes over the binary alphabet, soft decision decoding amounts to list decodability, which we focus on here. We give good list-decodable codes without using the composition. That is, by just applying our almost ℓ-wise uniform sampler on the binary alphabet, we can get a binary list-decodable code with rate $\Omega(\varepsilon^{2+\alpha})$ and non-trivial list size, although the list size is still exponential.

Theorem 4. *For every constant $\alpha > 0$: there exists an explicit binary code with rate $\Omega(\varepsilon^{2+\alpha})$ that is $(\rho = \frac{1}{2} - \varepsilon, L)$ list-decodable with list size $L = 2^{(1-c)n}$ for some constant $c = c(\alpha) > 0$.*

Prior to our work, the best known code rate was $\Omega(\varepsilon^3)$ by Guruswami and Rudra [GR08]. We emphasize that their code achieved a list size of $L = \mathrm{poly}(n)$, while our list size is exponentially large, making our code unlikely to be useful.

2 Preliminaries

Notations. We use $[t]$ to represent set $\{1, \ldots, t\}$. For integer m, U_m is a random variable distributed uniformly over $\{0,1\}^m$. For random variables X and Y, we use $X \approx_\varepsilon Y$ to represent the statistical distance (total variation distance) between X and Y is at most ε. We use the term "efficient" to mean polynomial-time computable.

2.1 Extractor-Based Sampler

As mentioned above, averaging samplers are equivalent to extractors. We will introduce this in detail in Section 4.1. Reingold, Vadhan, and Wigderson used this equivalence to achieve the following:

Theorem 2.1 ([RVW00, Corollary 7.3], see also [Gol11, Theorem 6.1])**.** *For every constant $\alpha > 0$, there exists an efficient (δ, ε) averaging sampler over $\{0,1\}^m$ with $\mathrm{poly}(1/\varepsilon, \log(1/\delta))$ samples using $m + (1+\alpha) \cdot \log_2(1/\delta)$ random bits.*

For ease of presentation, we often denote an extractor-based averaging sampler by $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$, where $\mathsf{Ext}(X, i)$ is the i-th output sample point of the sampler using randomness input X. Therefore, the sample complexity of Ext is 2^d.

2.2 Almost ℓ-wise Uniform

An almost ℓ-wise uniform sequence is a sequence of random variables such that the marginal distribution of every ℓ of them is close to uniform.

Definition 2.2 ([NN93])**.** *A sequence of random variables $Z_1, \ldots, Z_t \in \{0,1\}^m$ is said to be γ-almost ℓ-wise uniform if for all subsets $S \subseteq [t]$ such that $|S| \leq k$,*

$$(Z_i)_{i \in [S]} \approx_\gamma U_{m \times |S|}.$$

Lemma 2.3 ([NN93])**.** *There exists an efficient algorithm that uses $O(\ell m + \log(1/\gamma) + \log\log t)$ random bits to generate a γ-almost ℓ-wise uniform sequence $z_1, \ldots, z_t \in \{0,1\}^m$.*

Using standard techniques, we have the following concentration bound for almost ℓ-wise uniform sequences (see Appendix A for the proof). Bellare and Rompel [BR94] derived a similar bound for exact ℓ-wise uniform sequences.

Lemma 2.4. *Let $Z_1, \dots, Z_t \in \{0,1\}^m$ be a sequence of γ-almost ℓ-wise uniform variables for an even integer ℓ. Then for every sequence of functions $f_1, \dots, f_t : \{0,1\}^m \to [0,1]$,*

$$\Pr\left[\left|\frac{1}{t}\sum_{i=1}^{t}(f_i(Z_i) - \mathbb{E} f_i)\right| \le \varepsilon\right] \ge 1 - \left(\frac{5\sqrt{\ell}}{\varepsilon\sqrt{t}}\right)^{\ell} - \frac{\gamma}{\varepsilon^{\ell}}.$$

2.3 Composition of Samplers

The idea of composing samplers has been studied before. More specifically, Goldreich proved the following proposition.

Proposition 2.5 ([Gol11]). *Suppose we are given two efficient samplers:*

- *A (δ, ε) averaging sampler for domain $\{0,1\}^m$ with t_1 samples using n_1 random bits.*
- *A (δ', ε') averaging sampler for domain $\{0,1\}^{\log t_1}$ with t_2 samples using n_2 random bits.*

Then, there exists an efficient $(\delta + \delta', \varepsilon + \varepsilon')$ averaging sampler for domain $\{0,1\}^m$ with t_2 samples using $O(n_1 + n_2)$ random bits.

3 Main Results

Our construction is based on a reduction lemma that constructs a sampler for domain $\{0,1\}^m$ based on a sampler for domain $\{0,1\}^{O(\log(1/\varepsilon)+\log\log(1/\delta))}$. We exploit the fact that when composing averaging samplers, the final sample complexity depends on only one of the samplers. Our strategy is:

- Apply the extractor sampler in Theorem 2.1 as a $(\delta/2, \varepsilon/2)$ sampler over domain $\{0,1\}^m$. This uses $m + O(\log(1/\delta))$ random bits and generates $\mathrm{poly}(1/\varepsilon, \log(1/\delta))$ samples.
- By Proposition 2.5, we only need to design a $(\delta/2, \varepsilon/2)$ averaging sampler over domain $\{0,1\}^{O(\log(1/\varepsilon)+\log\log(1/\delta))}$ using $O(\log(1/\delta))$ random bits. The total sample complexity will be equal to the sample complexity of this sampler. For this sampler, we use almost ℓ-wise uniformity.

To formally prove the reduction lemma, we establish the next lemma, which demonstrates the explicit composition of samplers and proves that this composition maintains the properties of a strong sampler. The proof follows from the idea of Proposition 2.5.

Lemma 3.1. *Let $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ be a (δ, ε) averaging sampler, and let Samp_{base} be a (δ', ε') averaging sampler for domain $\{0,1\}^d$. Suppose $Y_1, \dots, Y_t \in \{0,1\}^d$ are the samples generated by Samp_{base}, i.e., for uniformly random source R,*

$$\mathsf{Samp}_{base}(R) = (Y_1, \dots, Y_t).$$

Then, for a uniformly random $X \in \{0,1\}^n$,

$$\mathsf{Samp}(R, X) := (\mathsf{Ext}(X, Y_1), \dots, \mathsf{Ext}(X, Y_t))$$

is a $(\delta' + \delta, \varepsilon' + \varepsilon)$ averaging sampler for domain $\{0,1\}^m$. Furthermore, if Samp_{base} is strong, then Samp is also a strong $(\delta' + t\delta, \varepsilon' + \varepsilon)$ averaging sampler.

Proof. We only prove the case when Samp_{base} is a strong sampler, and the non-strong case follows similarly. Let $f_1, \ldots, f_t : \{0,1\}^m \to [0,1]$ be an arbitrary sequence of functions. By the definition of strong samplers, we have for every f_i,

$$\Pr_{X \sim U_n}\left[\left|\mathop{\mathbb{E}}_{Y \sim U_d} f_i(\mathsf{Ext}(X,Y)) - \mathbb{E} f_i\right| \le \varepsilon\right] \ge 1 - \delta.$$

By a union bound over all $f_1, \ldots, f_t$, we have [1]

$$\Pr_{X \sim U_n}\left[\forall i \in [t] : \left|\mathop{\mathbb{E}}_{Y \sim U_d} f_i(\mathsf{Ext}(X,Y)) - \mathbb{E} f_i\right| \le \varepsilon\right] \ge 1 - t\delta. \tag{1}$$

For an arbitrary x, view $f_i(\mathsf{Ext}(x,\cdot))$ as a Boolean function on domain $\{0,1\}^d$. Therefore, since $Y_1, \ldots, Y_t$ are generated by a strong (δ, ε) sampler,

$$\Pr_{Y_1,\ldots,Y_t}\left[\left|\frac{1}{t}\sum_{i=1}^{t}\left(f_i(\mathsf{Ext}(x,Y_i)) - \mathop{\mathbb{E}}_{Y \sim U_d} f_i(\mathsf{Ext}(x,Y))\right)\right| \le \varepsilon'\right] \ge 1 - \delta'. \tag{2}$$

By the triangle inequality and a union bound over equations (1) and (2), we have

$$\Pr_{X,Y_1,\ldots Y_t}\left[\left|\frac{1}{t}\sum_{i=1}^{t}(f_i(\mathsf{Ext}(X,Y_i)) - \mathbb{E} f_i)\right| \le \varepsilon' + \varepsilon\right] \ge 1 - \delta' - t\delta.$$

This proves that $(\mathsf{Ext}(X,Y_1), \ldots, \mathsf{Ext}(X,Y_t))$ is a strong $(\delta' + t\delta, \varepsilon' + \varepsilon)$ averaging sampler. ☐

Instantiating Lemma 3.1 with the extractor-based sampler from Theorem 2.1 gives:

Lemma 3.2 (Main Reduction Lemma)**.** *For any $\alpha > 0$: For a sufficiently large constant $C > 0$, suppose there exists an efficient (δ', ε') averaging sampler Samp_{base} for domain $\{0,1\}^{C(\log(1/\varepsilon)+\log\log(1/\delta))}$ with t samples using n random bits. Then*

- *There exists an efficient $(\delta + \delta', \varepsilon + \varepsilon')$ averaging sampler Samp for domain $\{0,1\}^m$ with t samples using $m + (1+\alpha)\log(1/\delta) + n$ random bits.*
- *If Samp_{base} is strong, then there exists an efficient $(\delta + \delta', \varepsilon + \varepsilon')$ averaging sampler Samp for domain $\{0,1\}^m$ with t samples using $m + (1+\alpha)\log(t/\delta) + n$ random bits.*

Proof. By Theorem 2.1, there exists an explicit $(\delta/t, \varepsilon)$ averaging sampler $\mathsf{Ext} : \{0,1\}^{n'} \times \{0,1\}^d \to \{0,1\}^m$ with $n' = m + (1+\alpha)(\log(t/\delta))$ and $d = \log(\mathrm{poly}(1/\varepsilon, \log(t/\delta))) \le \frac{C}{2}(\log\log(t/\delta) + \log(1/\varepsilon))$ for some large enough constant C.

First, we need to verify that Samp_{base} can work for domain $\{0,1\}^d$, i.e., that $d \le C(\log(1/\varepsilon) + \log\log(1/\delta))$. Without loss of generality, we assume $\log t \le C(\log(1/\varepsilon) + \log\log(1/\delta))$; otherwise, we can just use the trivial sampler that outputs the whole domain. Thus, Samp_{base} can successfully work for $\{0,1\}^d$ since

$$d \le \frac{C}{2}(\log(1/\varepsilon) + \log\log(1/\delta)) + \frac{C}{2}\log\log t \le C(\log(1/\varepsilon) + \log\log(1/\delta))$$

Next, we analyze the number of random bits that Samp needs. We need n' random bits to generate x and we need n bits to generate $y_1, \ldots, y_t$.

Therefore, the total number of random bits we need is $n' + n = m + (1+\alpha)\log(t/\delta) + n$. The lemma then follows from Lemma 3.1. ☐

[1] Note that for the non-strong case, we don't need a union bound here. Thus, we can save a factor of t in the error parameter.

Next, we show that for domain $\{0,1\}^m$ with $m \le O(\log(1/\varepsilon) + \log\log(1/\delta))$, we can use an almost ℓ-wise uniform sequence to design a strong averaging sampler with near-optimal sample complexity.

Lemma 3.3. *For any constant $\alpha > 0$, there exists an efficient strong (δ, ε) averaging sampler for domain $\{0,1\}^m$ with $O(\frac{1}{\varepsilon^{2+\alpha}} \log \frac{1}{\delta})$ samples using $O(\frac{m \log(1/\delta)}{\log(1/\varepsilon)} + \log(1/\delta))$ random bits.*

Proof. We begin by setting $\ell = \frac{2\log(2/\delta)}{\alpha \log(1/\varepsilon)}$, $\gamma = \frac{\delta\varepsilon^\ell}{2}$, and $t = \frac{50\log(1/\delta)}{\alpha\varepsilon^{2+\alpha}\log(1/\varepsilon)}$. We then define our sampler by outputting a γ-almost ℓ-wise uniform sequence $Z_1, \dots, Z_t \in \{0,1\}^m$. Taking the parameters of Lemma 2.4, observe

$$\left(\frac{5\sqrt{\ell}}{\varepsilon\sqrt{t}}\right)^\ell = (\varepsilon^{\alpha/2})^{\frac{2}{\alpha}\log_\varepsilon \frac{\delta}{2}} = \frac{\delta}{2},$$

and

$$\frac{\gamma}{\varepsilon^\ell} = \frac{\delta}{2}.$$

Therefore, for every sequence of functions $f_1, \dots, f_t : \{0,1\}^m \to [0,1]$,

$$\Pr\left[\left|\frac{1}{t}\sum_{i=1}^t (f_i(Z_i) - \mathbb{E} f_i)\right| \le \varepsilon\right] \ge 1 - \delta.$$

Furthermore, Lemma 2.3 shows that we have an efficient algorithm that uses only $O(\frac{m\log(1/\delta)}{\log(1/\varepsilon)} + \log(1/\delta))$ random bits to generate this γ-almost ℓ-wise uniform sequence. □

Combining Lemma 3.2 and Lemma 3.3, we prove the main theorem.

Proof of Theorem 1. By Lemma 3.2, our goal is to design an efficient strong $(\delta/2, \varepsilon/2)$ averaging sampler Samp_{base} for domain $\{0,1\}^{C(\log(1/\varepsilon)+\log\log(1/\delta))}$ for some large enough constant C. The theorem is proved if Samp_{base} generates $O(\frac{1}{\varepsilon^{2+\alpha}}\log\frac{1}{\delta})$ samples using $O(\frac{\log(1/\delta)\log\log(1/\delta)}{\log(1/\varepsilon)} + \log(1/\delta))$ random bits. The almost ℓ-wise uniform sampler defined in Lemma 3.3 for $m = C(\log(1/\varepsilon) + \log\log(1/\delta))$ satisfies these conditions, and proves the theorem. □

Remark 3.4. Instead of using an almost ℓ-wise uniform sequence, we can also use a perfectly ℓ-wise uniform sequence to establish Theorem 1. Specifically, an ℓ-wise uniform sequence would give a strong (δ, ε) averaging sampler with $O(\frac{1}{\varepsilon^{2+\alpha}}\log\frac{1}{\delta})$ samples using $O(\frac{m\log(1/\delta)}{\log(1/\varepsilon)} + \log(1/\delta) + \frac{\log(1/\delta)\log\log(1/\delta)}{\log(1/\varepsilon)})$ random bits. This matches the bound of the almost ℓ-wise uniform sampler in Lemma 3.3 when $m = \Theta(\log(1/\varepsilon) + \log\log(1/\delta))$. However, it performs poorly for small domains, making it unable to yield Theorem 4.

4 Applications to Extractors and Codes

4.1 Applications to Extractors

Zuckerman showed that averaging samplers are equivalent to randomness extractors [Zuc97]. Here we state the only direction that we need.

Lemma 4.1 ([Zuc97])**.** *An efficient strong (δ, ε) averaging sampler $\mathsf{Samp} : \{0,1\}^n \to (\{0,1\}^m)^t$ gives an efficient strong $(n - \log(1/\delta) + \log(1/\varepsilon), 2\varepsilon)$ extractor $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^{\log t} \to \{0,1\}^m$.*

Applying Lemma 4.1 on Corollary 2 gives Theorem 3.

4.2 Application to List-Decodable Codes

Error-correcting codes are combinatorial objects that enable messages to be accurately transmitted, even when parts of the data get corrupted. Codes have been extensively studied and have proven to be extremely useful in computer science. Here we focus on the combinatorial property of list-decodability, defined below.

Definition 4.2. *A code* $\mathsf{ECC} : \{0,1\}^n \to (\{0,1\}^m)^t$ *is* (ρ, L) *list-decodable if for every received message* $r \in (\{0,1\}^m)^t$*, there are at most* L *messages* $x \in \{0,1\}^n$ *such that* $d_H(\mathsf{ECC}(x), r) \leq \rho t$*, where* d_H *denotes the Hamming distance. A code is binary if* $m = 1$*.*

We focus on the binary setting, i.e., $m = 1$.

Lemma 4.3 ([TZ04])**.** *An efficient strong* (δ, ε) *averaging sampler* $\mathsf{Samp} : \{0,1\}^n \to \{0,1\}^t$ *over the binary domain gives an efficient binary code that is* $(\rho = \frac{1}{2} - \varepsilon, \delta 2^n)$ *list-decodable with code rate* $R = n/t$*.*

Applying Lemma 4.3 to our almost ℓ-wise uniform sampler in Lemma 3.3 gives Theorem 4.

5 Open Problems

Our work raises interesting open problems.

- Can we remove the $\log(1/\delta) < \varepsilon^{-C}$ requirement in Corollary 2? Specifically, for constant ε and exponentially small δ, can we design a (δ, ε) averaging sampler with $O(\log(1/\delta))$ samples using $O(m + \log(1/\delta))$ random bits?
- Is it possible to reduce the list size of the list-decodable codes in Theorem 4 to $\mathrm{poly}(n)$ b the structure of the list?
- Comparing to the sampler in [RVW00] which uses $m + (1 + \alpha)\log(1/\delta)$ random bits, our construction requires $m + O(\log(1/\delta))$ random bits. Can we improve our randomness efficiency while maintaining a good sample complexity?
- Is there a way to eliminate the ε^α factor in the sample complexity? For $\varepsilon = 1/\mathrm{poly}(m)$ and $\delta = \exp(-\mathrm{poly}(m))$, can we design an efficient averaging sampler that is asymptotically optimal in both randomness and sample complexity? This will fully answer the open question raised by Bellare and Rompel.

Acknowledgements

We thank Oded Goldreich and Dana Moshkovitz for helpful comments.

Bibliography

[ALM+98] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and the hardness of approximation problems. *J. ACM*, 45(3):501–555, may 1998.

[BBR88] Charles H Bennett, Gilles Brassard, and Jean-Marc Robert. Privacy amplification by public discussion. *SIAM journal on Computing*, 17(2):210–229, 1988.

[BGG93] Mihir Bellare, Oded Goldreich, and Shafi Goldwasser. Randomness in interactive proofs. *Computational Complexity*, 3:319–354, 1993.

[BR94] Mihir Bellare and John Rompel. Randomness-efficient oblivious sampling. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pages 276–287. IEEE, 1994.

[CEG95] Ran Canetti, Guy Even, and Oded Goldreich. Lower bounds for sampling algorithms for estimating the average. *Information Processing Letters*, 53(1):17–25, 1995.

[CG89] Benny Chor and Oded Goldreich. On the power of two-point based sampling. *Journal of Complexity*, 5(1):96–106, 1989.

[Gil98] David Gillman. A chernoff bound for random walks on expander graphs. *SIAM Journal on Computing*, 27(4):1203–1220, 1998.

[Gol11] Oded Goldreich. A sample of samplers: A computational perspective on sampling. In *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*, pages 302–332. Springer, 2011.

[GR08] Venkatesan Guruswami and Atri Rudra. Explicit codes achieving list decoding capacity: Error-correction with optimal redundancy. *IEEE Transactions on Information Theory*, 54(1):135–150, 2008.

[HILL99] Johan Håstad, Russell Impagliazzo, Leonid A Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999.

[IZ89] Russell Impagliazzo and David Zuckerman. How to recycle random bits. In *FOCS*, volume 30, pages 248–253, 1989.

[Jus72] Jørn Justesen. Class of constructive asymptotically good algebraic codes. *IEEE Transactions on information theory*, 18(5):652–656, 1972.

[NN93] Joseph Naor and Moni Naor. Small-bias probability spaces-efficient constructions and applications. *SIAM Journal on Computing*, 22(4):838–856, 1993.

[NZ96] Noam Nisan and David Zuckerman. Randomness is linear in space. *Journal of Computer and System Sciences*, 52(1):43–52, 1996.

[Raz05] Ran Raz. Extractors with weak random seeds. In *Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, pages 11–20, 2005.

[RL01] Yao-Feng Ren and Han-Ying Liang. On the best constant in Marcinkiewicz–Zygmund inequality. *Statistics & probability letters*, 53(3):227–233, 2001.

[RT00] Jaikumar Radhakrishnan and Amnon Ta-Shma. Bounds for dispersers, extractors, and depth-two superconcentrators. *SIAM Journal on Discrete Mathematics*, 13(1):2–24, 2000.

[RVW00] Omer Reingold, Salil Vadhan, and Avi Wigderson. Entropy waves, the zig-zag graph product, and new constant-degree expanders and extractors. In *Proceedings 41st Annual Symposium on Foundations of Computer Science*, pages 3–13. IEEE, 2000.

[SZ99] Aravind Srinivasan and David Zuckerman. Computing with very weak random sources. *SIAM Journal on Computing*, 28(4):1433–1459, 1999.

[TZ04] Amnon Ta-Shma and David Zuckerman. Extractor codes. *IEEE transactions on information theory*, 50(12):3015–3025, 2004.

[Vad07] Salil Vadhan. The unified theory of pseudorandomness: guest column. *ACM SIGACT News*, 38(3):39–54, 2007.

[Zuc97] David Zuckerman. Randomness-optimal oblivious sampling. *Random Structures & Algorithms*, 11(4):345–367, 1997.

[Zuc07] David Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. *Theory of Computing*, 3(6):103–128, 2007.

A Proof of Lemma 2.4

Proposition A.1 (Marcinkiewicz–Zygmund inequality [RL01])**.** *Let $\{X_i, i \geq 1\}$ be a sequence of independent random variables with $\mathbb{E}\, X_i = 0$, $\mathbb{E}|X_i|^p < \infty$. Then for $p \geq 2$:*

$$\mathbb{E}\left|\sum_{i=1}^{n} X_i\right|^p \leq C(p) n^{p/2-1} \sum_{i=1}^{n} \mathbb{E}|X_i|^p,$$

where $C(p) \leq (3\sqrt{2})^p p^{p/2}$.

Proof of Lemma 2.4. Let $W_i := f_i(Z_i) - \mathbb{E}\, f_i$. We have

$$\Pr\left[\left|\sum_{i=1}^{t} W_i\right| > t\varepsilon\right] \leq \frac{\mathbb{E}\left[\left|\sum_{i=1}^{t} W_i\right|^\ell\right]}{(t\varepsilon)^\ell}.$$

Let $W_1', \ldots, W_t'$ be a sequence of independent random variables where $W_i' := f_i(U_{\{0,1\}^m}) - \mathbb{E}\, f_i$. Since the W_i's are γ-almost ℓ-wise independent and $|W_i| \leq 1$, we have

$$\mathbb{E}\left[\left|\sum_{i=1}^{t} W_i\right|^\ell\right] = \mathbb{E}\left[\left(\sum_{i=1}^{t} W_i\right)^\ell\right] \leq \mathbb{E}\left[\left(\sum_{i=1}^{t} W_i'\right)^\ell\right] + \gamma t^\ell = \mathbb{E}\left[\left|\sum_{i=1}^{t} W_i'\right|^\ell\right] + \gamma t^\ell.$$

Since $\mathbb{E}\, W_i' = 0$ and $|W_i'| \leq 1$. they satisfy the conditions for Marcinkiewicz–Zygmund inequality. We have

$$\mathbb{E}\left[\left|\sum_{i=1}^{t} W_i'\right|^\ell\right] \leq (3\sqrt{2})^\ell \ell^{\ell/2} t^{\ell/2-1} \sum_{i=1}^{t} \mathbb{E}|W_i'|^\ell \leq (5\sqrt{\ell t})^\ell.$$

Therefore,

$$\frac{\mathbb{E}\left[\left|\sum_{i=1}^{t} W_i\right|^\ell\right]}{(t\varepsilon)^\ell} \leq \left(\frac{5\sqrt{\ell}}{\varepsilon\sqrt{t}}\right)^\ell + \frac{\gamma}{\varepsilon^\ell}.$$

□

ECCC ISSN 1433-8092
https://eccc.weizmann.ac.il