A Sample of Samplers: A Computational Perspective on Sampling

Oded Goldreich
Department of Computer Science and Applied Mathematics
Weizmann Institute of Science, Rehovot, Israel.
Email: oded@wisdom.weizmann.ac.il

May 26, 1997

Abstract

We consider the problem of estimating the average of a huge set of values. That is, given oracle access to an arbitrary function $f : \{0,1\}^n \mapsto [0,1]$, we need to estimate $2^{-n} \sum_{x\in\{0,1\}^n} f(x)$ upto an additive error of ϵ. We are allowed to employ a randomized algorithm which may err with probability at most δ.

We survey known algorithms for this problem and focus on the ideas underlying their construction. In particular, we present an algorithm which makes $O(\epsilon^{-2} \cdot \log(1/\delta))$ queries and uses $n+O(\log(1/\epsilon))+O(\log(1/\delta))$ coin tosses, both complexities being very close to the corresponding lower bounds.

Keywords: Sampling, randomness complexity, saving randomness, pairwise independent random variables, Expander graphs, random walks on graphs, lower bounds.

1 Introduction

In many settings repeated sampling is used to estimate the average value of a huge set of values. Namely, there is a value function ν defined over a huge space, say $\nu : \{0,1\}^n \mapsto [0,1]$, and one wishes to approximate $\bar{\nu} \stackrel{\rm def}{=} \frac{1}{2^n}\sum_{x\in\{0,1\}^n} \nu(x)$ without having to inspect the value of ν on the entire domain. We comment that it is essential to have the range of ν be bounded (or else no reasonable approximation may be possible). Our convention of having $[0,1]$ be the range of ν is adopted for simplicity, and the problem for other (predetermined) ranges can be treated analogously.

1.1 Formal Setting

Our notion of approximation depends on two parameters: *accuracy* (denoted ϵ) and *error probability* (denoted δ). We wish to have an algorithm which with probability at least $1-\delta$, gets within ϵ of the correct value. This leads to the following definition.

Definition 1.1 (sampler): *A* sampler *is a randomized algorithm that on input parameters n* (length)*, ϵ* (accuracy) *and δ* (error)*, and oracle access to* any *function $\nu : \{0,1\}^n \mapsto [0,1]$, outputs, with probability at least $1-\delta$, a value that is at most ϵ away from $\bar{\nu} \stackrel{\rm def}{=} \frac{1}{2^n}\sum_{x\in\{0,1\}^n} \nu(x)$. Namely,*

$$\Pr(|\mathrm{sampler}^{\nu}(n,\epsilon,\delta) - \bar{\nu}| > \epsilon) \;<\; \delta$$

where the probability is taken over the internal coin tosses of the sampler.

We are interested in "the complexity of sampling" quantified as a function of the parameters n, ϵ and δ. Specifically, we will consider three complexity measures

1. Sample Complexity: the number of oracle queries made by the sampler.
2. Randomness Complexity: the number of (unbiased) coin tosses performed by the sampler.
3. Computational Complexity: the running-time of the sampler. We say that a sample is efficient if its running-time is polynomial in the total length of its queries (i.e., polynomial in both its sample complexity and in n).

We will focus on efficient samplers. Furthermore, we will focus on efficient samplers which have optimal (upto a constant factor) sample complexity, and will be interested in having the randomness complexity be as low as possible.

1.2 Overview

The straightforward method (or the naive sampler) consists of *uniformly and independently* selecting sufficiently many sample points (queries), and outputting the average value of the function on these points. Using Chernoff Bound one easily determines that $O(\frac{\log(1/\delta)}{\epsilon^2})$ sample points suffice. The naive sampler is optimal (upto a constant factor) in its sample complexity, but is quite wasteful in randomness. In Section 2, we discuss the naive sampler and present lower (and upper) bounds on the sample and randomness complexities of samplers. These will guide our quest for improvements.

Pairwise-independent sampling yields a great saving in the randomness complexity. In Section 3 we present the Pairwise-Independent Sampler, and discuss its advantages and disadvantages. Specifically, for constant $\delta > 0$, the Pairwise-Independent Sampler is optimal upto a constant factor in both its sample and randomness complexities. However, for small δ (i.e., $\delta = o(1)$), its sample complexity is wasteful.

A new idea is required for going further, and a relevant tool – random walks on expander graphs (see Appendix A) – is needed too. In Section 4, we combine the Pairwise-Independent Sampler with the Expander Random Walk Technique to obtain a new sampler. Loosely speaking, the new sampler uses a random walk on an expander to generate a sequence of $\ell \stackrel{\text{def}}{=} O(\log(1/\delta))$ (related) random pads for ℓ invocations of the Pairwise-Independent Sampler. Each of these invocations returns an ϵ-close approximation with probability at least 0.9. The expander walk technique yields that, with probability at least $1 - \exp(-\ell) = 1 - \delta$, most of these ℓ invocations return an ϵ-close approximation. Thus, the median value is an (ϵ, δ)-approximation to the correct value. The resulting sampler, called the Median-of-Averages Sampler, has sample complexity $O(\frac{\log(1/\delta)}{\epsilon^2})$ and randomness complexity $2n + O(\log(1/\delta))$.

In Section 5, we present an alternative sampler which improves over the pairwise-independent sampler. Maintaining the sample complexity of the latter (i.e., $O(1/\delta\epsilon^2)$), the new sampler has randomness complexity $n + O(\log(1/\delta\epsilon))$ (rather than $2n$). Combining this new sampler with the Expander Random Walk Technique, we obtain sample complexity $O(\frac{\log(1/\delta)}{\epsilon^2})$ and randomness complexity $n + O(\log(1/\delta)) + O(\log(1/\epsilon))$. Better bounds are obtained for the case of "Boolean samplers" (i.e., algorithms which must only well-approximate Boolean functions). In addition, in Section 5 we present two general techniques for refining samplers.

We conclude with some open problems. In particular, we discuss the notion of "oblivious" (or "averaging") samplers.

The Hitting Problem is considered in Appendix C. Here, given an oracle to a function having at least an ϵ fraction of 1's, one is required to find an input which evaluates to 1. Clearly, each sampler can be used for this purpose, but this is an over-kill. Nevertheless, all results and techniques for samplers (presented in this survey) have simpler analogies for the hitting problem. Thus, Appendix C can be read as a warm-up towards the rest of the survey.

2 The Information Theoretic Perspective

The Naive Sampler, presented below, corresponds to the information theoretical (or statistician) perspective of the problem. We augment it by a lower bound on the sampling complexity of samplers which is in the spirit of these areas. We conclude with lower and upper bounds on the randomness complexity of samplers: The latter lower bound is information theoretic in nature but represents a concern which is more common in computer science.

2.1 The Naive Sampler

The straightforward sampling method consists of randomly selecting a small sample set S and outputting $\frac{1}{|S|}\sum_{x\in S}\nu(x)$ as an estimate to $\bar{\nu}$. More accurately, we select m *independently and uniformly distributed* strings in $\{0,1\}^n$, denoted $s_1, ..., s_m$, and output $\frac{1}{m}\sum_{i=1}\nu(s_i)$ as our estimate. Setting $m = \frac{1+\ln(1/\delta)}{2\epsilon^2}$, we refer to this procedure as to the Naive Sampler.

To analyze the performance of the Naive Sampler, we use the Chernoff Bound. Specifically, we define m independent random variables, denoted $\zeta_1, ..., \zeta_m$, such that $\zeta_i \stackrel{\text{def}}{=} \nu(s_i)$, where the s_i's are independently and uniformly distributed in $\{0,1\}^n$. By Chernoff Bound:

$$\Pr\left(\left|\bar{\nu} - \frac{1}{m}\sum_{i=1}^{m}\zeta_i\right| > \epsilon\right) \leq 2\exp\left(-2\epsilon^2 m\right) \tag{1}$$

$$< \delta \tag{2}$$

where Eq. (2) is due to $m = (1+\ln(1/\delta))/(2\epsilon^2)$. Observing that $\frac{1}{m}\sum_{i=1}^{m}\zeta_i$ represents the estimate output by the Naive Sampler, we have established that the Naive Sampler indeed satisfies Definition 1.1 (i.e., is indeed a sampler). We now consider the complexity of the Naive Sampler

- Sample Complexity: $m \stackrel{\text{def}}{=} \frac{1+\ln(1/\delta)}{2\epsilon^2} = \Theta(\frac{log(1/\delta)}{\epsilon^2})$.
- Randomness Complexity: $m \cdot n = \Theta(\frac{log(1/\delta)}{\epsilon^2} \cdot n)$.
- Computational Complexity: indeed efficient.

In light of Theorem 2.1 (below), the sample complexity of the Naive Sampler is optimal upto a constant factor. However, as we will shortly see, it is extremely wasteful in its usage of randomness. In fact, the rest of this survey is devoted to presents ways for redeeming the latter aspect.

2.2 A Sample Complexity Lower Bound

We first assert that the Naive Sampler is quite good as far as sample complexity is concerned. The following theorem is analogous to many results known in statistics, though we are not aware of a prior reference where it can be found.

Theorem 2.1 [10]: *Any sampler has sample complexity bounded below by*

$$\min\left\{2^{(n-4)/2}, \frac{\ln(1/O(\delta))}{4\epsilon^2}\right\}$$

provided $\epsilon \leq \frac{1}{8}$ *and* $\delta \leq \frac{1}{6}$.

Note that a (constant factor) gap remains between the lower bound asserted here and the upper bound established by the Naive Sampler. We conjecture that the lower bound can be improved. Motivated by the lower bound, we say that a sampler is sample-optimal if its sample complexity is $O(\frac{\log(1/\delta)}{\epsilon^2})$.

2.3 Randomness Complexity Lower and Upper Bounds

We first assert that the Naive Sampler is quite bad as far as randomness complexity is concerned. First evidence towards our claim is provided by a non-explicit (and so inefficient) sampler:

Theorem 2.2 [10]: *There exists a* (non-efficient) *sampler with sample complexity* $\frac{1+\ln(1/\delta)}{2\epsilon^2}$ *and randomness complexity* $n + 2\log_2(2/\delta) + \log_2\log_2(1/\epsilon)$.

The proof is by a probabilistic argument which, given the Naive Sampler, asserts the existence of a relatively small set of possible coin tosses under which this sampler behaves almost as under all possible coin tosses (with respect to any possible function ν). Actually, the randomness bound can be improved to $n+\log_2(1/\delta)-\log_2\log_2(1/\delta)$ while using a constant factor larger sample complexity and more sophisticated techniques [27]. More generally,

Theorem 2.3 [27]: *For every function* $s : [0,1]^2 \mapsto \mathcal{R}$ *such that* $s(\epsilon, \delta) \geq \frac{2\log_2(1/\delta)}{\epsilon^2}$*, there exists a* (non-efficient) *sampler with sample complexity* $s(\epsilon, \delta)$ *and randomness complexity*

$$n + \log_2(1/\delta) + 2\log_2(4/\epsilon) - \log_2 s(\epsilon, \delta)$$

This gets us very close to the following lower bound

Theorem 2.4 [10]: *Let* $s : \mathcal{N} \times [0,1]^2 \mapsto \mathcal{R}$*. Any sampler which has sample complexity bounded above by* $s(n, \epsilon, \delta)$*, has randomness complexity bounded below by*

$$n + \log_2(1/\delta) - \log_2 s(n, \epsilon, \delta) - \log_2(1 - 2\epsilon)^{-1} - 2$$

provided $\epsilon, \delta < 0.5$ *and* $s(n, \epsilon, \delta) \leq 2^{n-1}$.

The dependency of the lower bound on the sample complexity should not come as a surprise. After all, there exist a deterministic sampler which just queries the function on the entire domain. Furthermore, the upper bound of Theorem 2.3 does express a similar trade-off between randomness complexity and sample complexity. Similarly, one should not be surprised at the effect of $1 - 2\epsilon$ on the bound: For example, when $\epsilon = 0.5$, a sample may merely output $\tilde{\nu} = \frac{1}{2}$ as its estimate and be within ϵ of the average of any function $\nu : \{0,1\}^n \mapsto [0,1]$.

Using Theorem 2.4, we obtain a lower bound on the randomness complexity of any *sample-optimal* sampler:

Corollary 2.5 [10]: *Any sampler which has sample complexity* $O(\frac{\log(1/\delta)}{\epsilon^2})$*, has randomness complexity bounded below by*

$$n + (1 - o(1)) \cdot \log_2(1/\delta) - 2\log_2(1/\epsilon)$$

provided $\epsilon, \delta < 0.4$ *and* $\frac{\log(1/\delta)}{\epsilon^2} = o(2^n)$.

The exact bound is $n + \log_2(1/\delta) - 2\log_2(1/\epsilon) - \log_2\log_2(1/\delta) - O(1)$.

3 The Pairwise-Independent Sampler

To motivate the Pairwise-Independent Sampler, let us confront two well-known central limit theorems: Chernoff Bound which refers to *totally independent* random variables and Chebishev's Inequality which refers to *pairwise-independent* random variables

Chernoff Bound: Let $\zeta_1, ..., \zeta_m$ be *totally independent* random variables, each ranging in $[0,1]$ and having expected value μ. Then,

$$\Pr\left(\left|\mu - \frac{1}{m}\sum_{i=1}^{m}\zeta_i\right| > \epsilon\right) \;\leq\; 2\exp\left(-2\epsilon^2 m\right)$$

Chebishev's Inequality: Let $\zeta_1, ..., \zeta_m$ be *pairwise-independent* random variables, each ranging in $[0,1]$ and having expected value μ. Then,

$$\Pr\left(\left|\mu - \frac{1}{m}\sum_{i=1}^{m}\zeta_i\right| > \epsilon\right) \;\leq\; \frac{1}{4\epsilon^2 m}$$

Our conclusion is that these two bounds essentially agree when $m = O(1/\epsilon^2)$. That is, in both cases $\Theta(1/\epsilon^2)$ identical random variables are necessary and sufficient to guarantee a concentration within ϵ with constant probability. Thus, if this is what we want then there is no point in using the more sophisticated Chernoff Bound which requires more of the random variables.

In the context of sampling, our conclusion is that for achieving an approximation to within ϵ accuracy with constant error probability, using $O(1/\epsilon^2)$ pairwise-independent random sample points is as good as using $O(1/\epsilon^2)$ totally independent random sample points. Furthermore, in the first case we may be save a lot in terms of randomness.

The Pairwise-Independent Sampler [13]: On input parameters n, ϵ and δ, set $m \stackrel{\rm def}{=} \frac{1}{4\epsilon^2\delta}$ and generate a sequence of m *pairwise-independently and uniformly distributed* strings in $\{0,1\}^n$, denoted $s_1, ..., s_m$. Using the oracle access to ν, output $\frac{1}{m}\sum_{i=1}\nu(s_i)$ as the estimate to $\bar{\nu}$. Using Chebishev's Inequality, one can easily see that the Pairwise-Independent Sampler indeed satisfies Definition 1.1 (i.e., is indeed a sampler).

There are two differences between the Naive Sampler and the Pairwise-Independent Sampler. Whereas the former uses independently selected sample points, the latter uses a sequence of pairwise independent sample points. As we'll see below, this allows the latter sampler to use much less randomness. On the other hand, the Naive Sampler uses $O(\frac{\log(1/\delta)}{\epsilon^2})$ samples (which is optimal upto a constant factor), whereas the Pairwise-Independent Sampler uses $O(\frac{1}{\epsilon^2\delta})$ samples. However, for constant δ, both samplers use essentially the same number of sample points. Thus, for constant δ, the Pairwise-Independent Sampler offers a saving in randomness while being sample-optimal.

Generating a Pairwise-Independent sequence: Whereas generating m totally independent random points in $\{0,1\}^n$ requires $m \cdot n$ unbiased coin flips, one can generate t pairwise-independent random points using only $O(n)$ unbiased coin flips. We present two well-known ways of doing this.

1. Linear functions over finite fields: We associate $\{0,1\}^n$ with the finite field $F \stackrel{\rm def}{=} {\rm GF}(2^n)$. Let $\alpha_1, ..., \alpha_m$ be $m \leq |F|$ distinct elements of F. To generate a (pairwise-independent) sequence of length m, we uniformly and independently selects $s, r \in F$, and let the $i^{\rm th}$ element in the sequence be $e_i \stackrel{\rm def}{=} r + \alpha_i s$ (where the arithmetic is that of F). The analysis of this construction "reduces" the stochastic independence of e_i and e_j to the linear independence of the vectors $(1\ ,\ \alpha_i)$ and $(1\ ,\ \alpha_j)$: For every $i \neq j$ and every $a, b \in F$, we have

$$\begin{aligned}
{\rm Pr}_{r,s}\left(e_i = a \ \wedge\ e_j = b\right) &= {\rm Pr}_{r,s}\left(\begin{pmatrix} 1 & \alpha_i \\ 1 & \alpha_j \end{pmatrix}\begin{pmatrix} r \\ s \end{pmatrix} = \begin{pmatrix} a \\ b \end{pmatrix}\right) \\
&= {\rm Pr}_{r,s}\left(\begin{pmatrix} r \\ s \end{pmatrix} = \begin{pmatrix} 1 & \alpha_i \\ 1 & \alpha_j \end{pmatrix}^{-1}\begin{pmatrix} a \\ b \end{pmatrix}\right) \\
&= \frac{1}{|F|^2}
\end{aligned}$$

Only $2n$ random coins are required in this construction, but the drawback is that we need a representation of the field F (i.e., an irreducible polynomial of degree n over GF(2)) which is not easy to find.[1]

[1] Things are not better if we wish to work with a large field of prime cardinality; since we need to find such a prime.

2. Toeplitz matrices: To avoid problems with non-trivial representation, one may use the following construction (which is Levin's favorite). We associate $\{0,1\}^n$ with the n-dimensional vector space over GF(2). Let $v_1, ..., v_m$ be $m \leq 2^n$ distinct vectors in this vector space. A Toeplitz matrix is a matrix with all diagonals being homogeneous; that is, $T = (t_{i,j})$ is a Toeplitz matrix if $t_{i,j} = t_{i+1,j+1}$, for all i,j. Note that a Toeplitz matrix is determined by its first row and first column (i.e., the values of $t_{1,j}$'s and $t_{i,1}$'s). To generate a (pairwise-independent) sequence of length m, we uniformly and independently select an n-by-n Toeplitz matrix, T, and an n-dimensional vector u. We let the i^{th} element in the sequence be $e_i \stackrel{\text{def}}{=} Tv_i + u$ (where the arithmetic is that of the vector space). The analysis of this construction is given in Appendix B. Here, we merely note that $3n-1$ random coins suffice for this construction,

Plugging-in either of these constructions, we obtain the following complexities for the Pairwise-Independent Sampler

- Sample Complexity: $\frac{1}{4\delta\epsilon^2}$.
- Randomness Complexity: $2n$ or $3n-1$, depending on which of the constructions is used.
- Computational Complexity: indeed efficient.

We note that for constant δ, the sample and randomness complexities match the lower bounds upto a constant factor. However, as δ decreases, the sample complexity of the Pairwise-Independent Sampler increases faster than the corresponding complexity of the Naive Sampler. Redeeming this state of affairs is our next goal.

4 The combined Median-of-Averages Sampler

Our goal here is to decrease the sample complexity of the Pairwise-Independent Sampler while essentially maintaining its randomness complexity. To motivate the new construction we first consider an oversimplified version of it.

Median-of-Averages Sampler (oversimplified): On input parameters n, ϵ and δ, set $m \stackrel{\text{def}}{=} \Theta(\frac{1}{\epsilon^2})$ and $\ell \stackrel{\text{def}}{=} \Theta(\log(1/\delta))$, generate ℓ independent m-element sequences, each being a sequence of m *pairwise-independently and uniformly distributed* strings in $\{0,1\}^n$. Denote the sample points in the i^{th} sequence by $s_1^i, ..., s_m^i$. Using the oracle access to ν, compute $\tilde{\nu}^i \stackrel{\text{def}}{=} \frac{1}{m}\sum_{j=1}^m \nu(s_j^i)$, for $i = 1, ..., \ell$, and output the *median value* among these $\tilde{\nu}^i$'s. Using Chebishev's Inequality (as in previous section), we have for each i

$$\Pr(|\tilde{\nu}^i - \bar{\nu}| > \epsilon) < 0.1$$

and so

$$\begin{aligned}
\Pr(|\{i : |\tilde{\nu}^i - \bar{\nu}| > \epsilon\}|) &< \sum_{j=\ell/2}^{\ell} \binom{\ell}{j} \cdot 0.1^j \cdot 0.9^{\ell-j} \\
&< 2^\ell \cdot 0.1^{\ell/2} \\
&\leq \delta
\end{aligned}$$

where the last inequality is due to the choice of ℓ. Thus, the oversimplified version described above is indeed a sampler and has the following complexities

- Sample Complexity: $\ell \cdot m = O(\frac{\log(1/\delta)}{\epsilon^2})$.
- Randomness Complexity: $\ell \cdot O(n) = O(n \cdot \log(1/\delta))$.
- Computational Complexity: indeed efficient.

Thus, the sample complexity is optimal (upto a constant factor), but the randomness complexity is higher than what we aim for. To reduce the randomness complexity, we use the same approach as above, but take dependent (pairwise independent) sequences rather than independent ones. The dependency we use is such which essentially preserves the probabilistic behavior of independent choices. Specifically, we use random walks on expander graphs (cf., Appendix A) to generate a sequence of ℓ "seeds" each of length $O(n)$. Each seed is used to generate a sequence of m pairwise independent elements in $\{0,1\}^n$, as above.

Theorem 4.1 (general median-composition [7]): *Suppose we are given an efficient sampler of sample complexity $s(n,\epsilon,\delta)$ and randomness complexity $r(n,\epsilon,\delta)$. Then,*

1. *there exists an efficient sampler with sample complexity $O(s(n,\epsilon,0.01)\cdot\log(1/\delta))$ and randomness complexity $r(n,\epsilon,0.01)+O(\log(1/\delta))$.*
2. *for any $c>4$, there exists an $\alpha>0$ and an efficient sampler with sample complexity $O(s(n,\epsilon,\alpha)\cdot\log(1/\delta))$ and randomness complexity $r(n,\epsilon,\alpha)+c\cdot\log_2(1/\delta)$.*

Proof: For Item 1, let $r \stackrel{\rm def}{=} r(n,\epsilon,0.01)$. We use an explicit construction of expander graphs with vertex set $\{0,1\}^r$, degree d and second eigenvalue λ so that $\lambda/d<0.1$. We consider a random walk of (edge) length $\ell-1=O(\log(1/\delta))$ on this expander, and use each of the ℓ vertices along the path as random coins for the given sampler. Thus, we obtain ℓ estimates to $\bar{\nu}$ and output the median value as the estimate of the new sampler. To analyze the performance of the resulting sampler, we let W denote the set of coin tosses (for the basic sampler) which make the basic sampler output an estimate which is ϵ-far from the correct value (i.e., $\bar{\nu}$). By the hypothesis, $\frac{|W|}{2^r}\leq 0.01$, and using Theorem A.5, the probability that at least $\ell/2$ vertices of the path reside in W is bounded above by

$$\begin{aligned}\sum_{j=\ell/2}^{\ell}\binom{\ell}{j}\cdot 0.2^j &< 2^\ell\cdot 0.2^{\ell/2}\\ &\leq \delta\end{aligned}$$

Note that we have used $\ell\cdot s(n,\epsilon,0.01)$ samples and $r+(\ell-1)\cdot\log_2 d = r+O(\log(1/\delta))$ coin tosses. Item 1 follows.

Item 2 is proven using the same argument but using Ramanujan Graphs and slightly more care. Specifically, we use Ramanujan graphs (i.e., expanders with $\lambda\leq 2\sqrt{d-1}$) with vertex set $\{0,1\}^r$, where $r\stackrel{\rm def}{=} r(n,\epsilon,\alpha)$ and $\alpha=(\frac{\lambda}{d})^2$. Repeating the above argument, with $\ell-1=\frac{2\log_2(1/\delta)}{\log_2(\alpha/8)}$, we obtain an efficient sampler which uses $\ell\cdot s(n,\epsilon,\alpha)$ samples and $r+(\ell-1)\cdot\log_2 d = r+(4+\frac{16}{(\log_2 d)-8})\cdot\log_2(1/\delta)$ coin tosses. Item 2 follows. ■

Combining the Pairwise-Independent Sampler with Theorem 4.1, we get

Corollary 4.2 (The Median-of-Averages Sampler [7]): *There exists an efficient sampler with*

- Sample Complexity: $O(\frac{\log(1/\delta)}{\epsilon^2})$.

- Randomness Complexity: $O(n + \log(1/\delta))$.

Furthermore, we can obtain randomness complexity $2n + (4 + o(1)) \cdot \log_2(1/\delta))$.

In the next section, we further reduce the randomness complexity of samplers to $n + O(\log(1/\epsilon) + \log(1/\delta))$, while maintaining the sample complexity (up-to a multiplicative constant).

5 The Expander Sampler and two Generic Techniques

The main result of this section is

Theorem 5.1 [7, 17]: *There exists an efficient sampler which has*

- Sample Complexity: $O(\frac{\log(1/\delta)}{\epsilon^2})$.
- Randomness Complexity: $n + \log_2(1/\epsilon) + O(\log(1/\delta))$.

The theorem is proven by applying Theorem 4.1 to a new efficient sampler which makes $O(\frac{1}{\delta\epsilon^2})$ oracle queries and tosses $n + \log_2(1/\epsilon)$ coins. We start by presenting a sampler for the special case of Boolean functions.

Definition 5.2 (Boolean sampler): *A* Boolean sampler *is a randomized algorithm that on input parameters n, ϵ and δ, and oracle access to* any Boolean *function* $\nu : \{0,1\}^n \mapsto \{0,1\}$, *outputs, with probability at least* $1-\delta$, *a value that is at most ϵ away from* $\bar{\nu} \stackrel{\rm def}{=} \frac{1}{2^n}\sum_{x\in\{0,1\}^n} \nu(x)$. *Namely,*

$$\Pr(|\text{sampler}^{\nu}(n,\epsilon,\delta) - \bar{\nu}| > \epsilon) \;<\; \delta$$

where the probability is taken over the internal coin tosses of the sampler.

That is, unlike (general) samplers, a Boolean Sampler is required to work well only when given access to a Boolean function. The rest of this section is organized as follows:

Subsection 5.1: The Expander Sampler – a Boolean sampler using Ramanujan Graphs.

Subsection 5.2: From Boolean samplers to general ones.

Subsection 5.3: The Expander Sampler, revisited. Here we use an arbitrary expander and a generic composition of samplers.

Theorem 5.1 is proven by combining the ideas of Subsections 5.1 and 5.2. An alternative proof of a somewhat weaker result is obtained by combining the ideas of Subsections 5.1 and 5.3.

5.1 A Sampler for the Boolean Case

We start by presenting a sampler for the special case of Boolean functions. Our sampling procedure is exactly the one suggested by Karp, Pippinger and Sipser for hitting a witness set [20] (cf., Appendix C), yet the analysis is somewhat more involved. Furthermore, to get an algorithm which samples the universe only on $O(1/\delta\epsilon^2)$ points, it is crucial to use a Ramanujan graph in role of the expander in the Karp-Pippinger-Sipser method.

The sampler. We use an expander of degree $d = 4/\delta\epsilon^2$ second eigenvalue bounded by λ and associate the vertex set of the expander with $\{0,1\}^n$. The sampler consists of uniformly selecting a vertex, v, (of the expander) and averaging over the values assigned (by ν) to all the neighbors of v; namely, the algorithm outputs the estimate

$$\tilde{\nu} \stackrel{\rm def}{=} \frac{1}{d} \sum_{u \in \mathcal{N}(v)} \nu(u)$$

where $\mathcal{N}(v)$ denotes the set of neighbors of vertex v. The algorithm has

- Sample Complexity: $O(\frac{1}{\delta\epsilon^2})$.
- Randomness Complexity: n.
- Computational Complexity: polynomial in n, ϵ^{-1} and δ^{-1}.

Lemma 5.3 [17]: *The above algorithm constitutes an efficient Boolean sampler.*

Proof: We denote by B the set of *bad* choices for the algorithm; namely, the set of vertices that once selected by the algorithm yield a wrong estimate. That is, $v \in B$ if

$$\left| \frac{1}{d} \sum_{u \in \mathcal{N}(v)} \nu(u) - \bar{\nu} \right| > \epsilon$$

Denote by B' the subset of $v \in B$ for which

$$\frac{1}{d} \sum_{u \in \mathcal{N}(v)} \nu(u) > \bar{\nu} + \epsilon \qquad (3)$$

It follows that each $v \in B'$ has ϵd too many neighbors in the set $A \stackrel{\rm def}{=} \{u : \nu(u)=1\}$; namely,

$$|\{u \in \mathcal{N}(v) : u \in A\}| > (\rho(A)+\epsilon)\cdot d \qquad (4)$$

where $\rho(A) \stackrel{\rm def}{=} \frac{|A|}{N}$ and $N \stackrel{\rm def}{=} 2^n$. Using the Expander Mixing Lemma (Lemma A.3) ones gets that

$$\begin{aligned} \epsilon \cdot \rho(B') &= \left| \frac{|B'|\cdot(\rho(A)+\epsilon)d}{dN} - \rho(B')\cdot\rho(A) \right| \\ &\leq \left| \frac{|(B' \times A) \cap E|}{|E|} - \frac{|A|}{|V|}\cdot\frac{|B'|}{|V|} \right| \\ &\leq \frac{\lambda}{d}\cdot\sqrt{\rho(A)\cdot\rho(B')} \end{aligned}$$

Thus,

$$\rho(B') \leq \left(\frac{\lambda}{d\epsilon}\right)^2 \cdot \rho(A) \qquad (5)$$

Using $\lambda \leq 2\sqrt{d}$ and $d = \frac{4}{\delta\epsilon^2}$, we get $\rho(B') \leq \delta\cdot\rho(A)$. Using a similar argument, we can show that $\rho(B \setminus B') \leq \delta\cdot(1-\rho(A))$. Thus, $\rho(B) \leq \delta$ and the claim follows. ■

Comment 5.4 [17]: *Observe that if we were to use an arbitrary d-regular graph with second eigenvalue λ then the above proof would hold provided that*

$$\frac{\lambda}{d} \leq \frac{\sqrt{\delta\epsilon^2}}{2} \qquad (6)$$

This would have yield an efficient Boolean sampler with sample complexity *d and* randomness complexity *n.*

5.2 From Boolean Samplers to General Samplers

The following generic transformation was suggested to us by Luca Trevisan.

Theorem 5.5 (Boolean samplers imply general ones): *Suppose we are given an efficient Boolean sampler of sample complexity $s(n,\epsilon,\delta)$ and randomness complexity $r(n,\epsilon,\delta)$. Then, there exists an efficient sampler with sample complexity $s(n+\log_2(1/\epsilon),\epsilon/2,\delta)$ and randomness complexity $r(n+\log_2(1/\epsilon),\epsilon/2,\delta)$.*

Proof: As a mental experiment, given an arbitrary function $\nu:\{0,1\}^n \mapsto [0,1]$, we define a Boolean function $\mu:\{0,1\}^{n+\ell} \mapsto [0,1]$, where $\ell \stackrel{\rm def}{=} \log_2(1/\epsilon)$, as follows: For $i=1,...,\epsilon^{-1}$, $\mu(x,i) \stackrel{\rm def}{=} 1$ if and only if $\nu(x) < (i-0.5)\cdot\epsilon$. Then, $|\nu(x) - \epsilon\cdot\sum_{i=1}^{1/\epsilon}\mu(x,i)| < \epsilon/2$. Thus, if we were to sample μ and obtain an $\epsilon/2$-approximation of $\bar{\mu}$ then we get an ϵ-approximation of $\bar{\nu}$. Now, although we don't have actual access to μ we can emulate its answers given an oracle to ν.

Given a Boolean sampler, B, we construct a general sampler, A, as follows. On input n,ϵ,δ and access to an arbitrary ν as above, algorithm A sets $n'=n+\ell$, $\epsilon'=\epsilon/2$, and $\delta'=\delta$, and invoke B on input n',ϵ',δ'. When B makes a query $(x,i)\in\{0,1\}^n\times\{0,1\}^\ell$, algorithm A queries for $\nu(x)$ and returns 1 if and only if $\nu(x)<(i-0.5)\cdot\epsilon$. When B halts with output v, A does the same. The theorem follows. ■

As a corollary to the above, we get

Corollary 5.6 *There exists an efficient sampler which has*

- Sample Complexity: $O(\frac{1}{\delta\epsilon^2})$.
- Randomness Complexity: $n+\log_2(1/\epsilon)$.

Theorem 5.1 follows by combining Corollary 5.6 with Theorem 4.1.

5.3 The Expander Sampler, Revisited

Using an arbitrary expander graph (with $d=\mathrm{poly}(1/\epsilon\delta)$ and $\frac{\lambda}{d}<\frac{\sqrt{\delta\epsilon^2}}{2}$) and invoking Comment 5.4, we have an efficient Boolean sampler with sample complexity $\mathrm{poly}(1/\epsilon\delta)$ and randomness complexity n. Using Theorem 5.5, we get

Corollary 5.7 *There exists an efficient sampler with* sample complexity $\mathrm{poly}(1/\epsilon\delta)$ *and* randomness complexity $n+\log_2(1/\epsilon)$.

To derive (a weaker form of) Theorem 5.1 via the above sampler, we first need to reduce its sample complexity. This is done via the following general transformation. We say that a sampler is of the **averaging type** if its output is the average value obtained on its queries which are determined as a function of its own coin tosses (independently of the answers obtained on previous queries).

Theorem 5.8 (reducing sample complexity): *Suppose we are given two efficient samplers so that the $i^{\rm th}$ sampler has sample complexity $s_i(n,\epsilon,\delta)$ and randomness complexity $r_i(n,\epsilon,\delta)$. Further suppose that the first sampler is of the averaging type. Then, there exists an efficient sampler with sample complexity $s_2(\log_2 s_1(n,\epsilon/2,\delta/2),\epsilon/2,\delta/2)$ and randomness complexity $r_1(n,\epsilon/2,\delta/2)+r_2(\log_2 s_1(n,\epsilon/2,\delta/2),\epsilon/2,\delta/2)$.*

Proof: We compose the two samplers as follows. Setting $m \stackrel{\text{def}}{=} s_1(n, \epsilon/2, \delta/2)$, we invoke the first sampler and determine the m queries it would have asked (given a particular choice of its coins).[2] We then use the second sampler to sample these m queries (invoking it with parameters $\log_2 m, \epsilon/2$ and $\delta/2$). That is, we let the second sampler make virtual queries into the domain $[m] \stackrel{\text{def}}{=} \{1, ..., m\}$ and answer a query $q \in [m]$ by the value of the function at the i^{th} query specified by the first sampler. That is, given access to a function $\nu : \{0,1\}^n \mapsto [0,1]$, and determining a sequence r of coins for the first sampler, we consider the function $\nu_r : [m] \mapsto [0,1]$ defined by letting $\nu_r(i) = \nu(q_{r,i})$ where $q_{r,i}$ is the i^{th} query made by the first sampler on coins r. We run the second sampler providing it virtual access to the function ν_r in the obvious manner, and output its output. Thus, the complexities are as claimed and the combined sampler errs if either $|\bar{\nu} - \frac{1}{m}\sum_{i=1}^m \nu(q_{r,i})| > \frac{\epsilon}{2}$ or $|\frac{1}{m}\sum_{i=1}^m \nu(q_{r,i}) - \tilde{\nu}_r| > \epsilon/2$, where $\tilde{\nu}_r$ is the estimate output by the second sampler when given virtual access to ν_r. Observing that the first event means that the first sampler errs (here we use the hypothesis that this sampler is averaging) and that the second event means that the second sampler errs (here we use $\sum_{i=1}^m \nu(q_{r,i}) = \bar{\nu}_r$), we are done. ■

Note that the sampler used to establish Corollary 5.7 is of the averaging type. Combining this sample with the Pairwise-Independent Sampler, via Theorem 5.8, we obtain

Corollary 5.9 *There exists an efficient sampler which has*

- Sample Complexity: $O(\frac{1}{\delta\epsilon^2})$.
- Randomness Complexity: $n + O(\log(1/\epsilon)) + O(\log(1/\delta))$.

A weaker form of Theorem 5.1 (i.e., $O(\log(1/\epsilon)$ rather than $\log_2(1/\epsilon)$ term) follows by combining Corollary 5.9 with Theorem 4.1.

6 Conclusions and Open Problems

The main results are summarized in Figure 1. The first row tabulates $\Omega(\epsilon^{-2}\log(1/\delta))$ as a lower bound on sample complexity and the subsequent three rows refer to sample-optimal samplers (i.e., samplers of sample complexity $O(\epsilon^{-2}\log(1/\delta)))$. The last row refers to a sampler (cf., Thm. 6.1 below) which, for $\delta < 2^{-n/O(1)}$, has randomness complexity closer to the lower bound. However, this sampler is not sample-optimal.

The randomness complexity of sample-optimal samplers: A closer look at the randomness complexity of sample-optimal samplers is provided in Figure 2. The first two rows tabulate lower and upper bounds which are $2\log_2(1/\epsilon) = O(1)$ apart. Our conjecture is that the lower bound can be improved to match the upper bound. The efficient samplers use at least $n + 4\log_2(1/\delta)$ coins, where one factor of 2 is due to the use of expanders and the other to the "median-of-averages paradigm". As long as we stick to using expanders in the Median-of-Averages Sampler, there is no hope to reduce the first factor which is due to the relation between the expander degree and its second eigenvalue. Actually, achieving a factor of 4 rather than a bigger factor is due to the use of Ramanujan Graphs (which have the best possible such relation).

[2] Here we use the hypothesis that the first sampler is non-adaptive; that is, its queries are determined by its coin tosses (independently of the answers obtained on previous queries).

	sample complexity	randomness complexity	pointer
lower bound	$\Omega(\frac{\log(1/\delta)}{\epsilon^2})$		Thm. 2.1
lower bound	for $O(\frac{\log(1/\delta)}{\epsilon^2})$	$n + (1 - o(1)) \cdot \log_2(1/\delta) - 2\log_2(1/\epsilon)$	Cor. 2.5
upper bound	$O(\frac{\log(1/\delta)}{\epsilon^2})$	$n + \log_2(1/\delta)$	Thm. 2.3
algorithm	$O(\frac{\log(1/\delta)}{\epsilon^2})$	$n + O(\log(1/\delta)) + \log_2(1/\epsilon)$	Thm. 5.1
algorithm	$\mathrm{poly}(n, \epsilon^{-1}, \log(1/\delta))$	$(1 + \alpha)(n + \log(1/\delta)), \forall\alpha > 0$	Thm. 6.1

Figure 1: Summary of results.

lower bound (even for Boolean)	$n + \log_2(1/\delta) - 2\log_2(1/\epsilon) - \log_2\log_2(1/\delta) - O(1)$
upper bound	$n + \log_2(1/\delta) - \log_2\log_2(1/\delta)$
efficient samplers	$n + (4 + \alpha)\log_2(1/\delta) + \log_2(1/\epsilon)$, for any $\alpha > 0$
efficient Boolean samplers	$n + (4 + \alpha)\log_2(1/\delta)$, for any $\alpha > 0$

Figure 2: The randomness complexity of samplers which make $\Theta(\frac{\log(1/\delta)}{\epsilon^2})$ queries.

Boolean Samplers vs general ones: Another fact presented in Figure 2 is that we can currently do better if we are guaranteed that the oracle function is Boolean (rather than mapping to the interval $[0, 1]$). We stress that the lower bound holds also with respect to samplers which need only to work for Boolean functions.

Adaptive vs non-adaptive: All known samplers are non-adaptive; that it, they determine the sample points (queries) as a function of their coin tosses. More general, *adaptive* samplers may determine the next query depending on the value of the function on previous queries. Intuitively, adaptivity should not help the sampler. Indeed, all lower bound refer also to adaptive samplers and so, in conjunction with the upper bound which only utilize non-adaptive samplers, indicate that the difference between adaptive samplers and non-adaptive ones can not be significant. Yet, it will be nice to have a direct and more tight proof of the above intuition.

Averaging (or Oblivious) Samplers: A special type of non-adaptive samplers are ones which output the average value of the function over their sample points. Such samplers were first defined in [9] and called "oblivious". We prefer the term *averaging*. Averaging samplers have some applications not offered by arbitrary non-adaptive samplers (cf., [9] and [25]). More importantly, averaging samplers are very appealing since averaging over a sample seem *the natural thing to do.* Furthermore, as pointed out in [27], averaging samplers are related to dispersers and to randomness extractors. Note that the Naive Sampler, the Pairwise-Independent Sampler and the Expander Sampler are all averaging samplers although they differ in the way they generate their sample. However, the Median-of-Averages Sampler, as its name indicates, is not an averaging sampler. Thus, to obtain an averaging sampler having relatively low sample and randomness complexities, an alternative approach is required. The best know result in this direction is:

Theorem 6.1 [27]: *For every $\alpha > 0$, there exists an efficient* averaging *sampler with sample complexity* $\mathrm{poly}(n, \epsilon^{-1}, \log(1/\delta))$ *and randomness complexity* $(1 + \alpha)(n + \log(1/\delta))$.

We stress that this sampler is not sample-optimal (i.e., the polynomial in ϵ^{-1} is of quite high degree). It would be interesting to obtain a sample-optimal averaging sampler of low randomness complexity, say, one which uses $O(n + \log(1/\delta))$ coins.

Sampling with weak sources: So far our discussion presupposed that a randomized algorithm has at its disposal a uniformly selected string of certain length (which may be considered the outcome of its internal coin tosses). A question which has received a lot of attention in the last decade is whether algorithms can be transformed into robust counterparts which may work given also "weak random sources" (cf., e.g., [26]). Following [12], we call a random variable X a (ℓ, m)-source if its support is a subset of $\{0,1\}^\ell$ and no string in its support is assigned probability mass greater than 2^{-m}. That is, m is a lower bound on the *min-entropy* of X defined as $\min_{\alpha\in\{0,1\}^\ell}\{-\log_2(Prob(X=\alpha))\}$. A recent result [6] states that any BPP-algorithm can be converted to work with very weak sources. Specifically, for any $\gamma > 0$, there exists a robust BPP-algorithm working with any (ℓ, ℓ^γ)-source. More generally,

Theorem 6.2 [6]: *For every $\gamma > 0$ and δ, there exists a polynomial p and a deterministic algorithm which for any n, ϵ and any $(p(n/\epsilon), p(n/\epsilon)^\gamma)$-source X, given input (n, ϵ, X) and access to any oracle $\nu : \{0,1\}^n \mapsto [0,1]$, runs in time* $\mathrm{poly}(n/\epsilon)$ *and outputs a value $\tilde{\nu}$ so that*

$$\Pr(|\tilde{\nu} - \bar{\nu}| > \epsilon) < 2^{-n^\delta}$$

Acknowledgments

I would like to thank Noga Alon, Nabil Kahale, and Luca Trevisan for useful discussions.

Dedication

The idea of writing this survey has first occurred to me when finding out that a brilliant, young researcher who works in very related areas was unaware of the Median-of-Averages Sampler. It has then occurred to me that many of the results presented above have appeared in papers devoted to other (more specific) subjects and have thus escaped the attention of a wider community which might have cared to know about them. Thus, I've decided to try to redeem the situation and it seems fair to dedicate my attempt to this researcher:

> *Forasmuch as many have taken in hand to set forth in order a declaration of those things which are most surely believed among us; Even as they delivered them unto us, who from the beginning were eyewitnesses, and ministers of the word; It seems good to me also, having had perfect understanding of all things from the very first, to write unto thee in order, most excellent Theophilus; That thou mightest know the certainty of those things, wherein thou hast been instructed.*
>
> Luke, 1:1-4, c. A.D. 60.

References

[1] M. Ajtai, J. Komlos, E. Szemerédi, "Deterministic Simulation in LogSpace", *Proc. 19th STOC*, 1987, pp. 132–140.

[2] N. Alon, "Eigenvalues, Geometric Expanders, Sorting in Rounds and Ramsey Theory", *Combinatorica*, 6 (1986), pp. 231–243.

[3] N. Alon, J. Bruck, J. Naor, M. Naor and R. Roth, "Construction of Asymptotically Good, Low-Rate Error-Correcting Codes through Pseudo-Random Graphs", *IEEE Transactions on Information Theory* **38** (1992), pp. 509–516.

[4] N. Alon and V.D. Milman, λ_1, Isoperimetric Inequalities for Graphs and Superconcentrators, *J. Combinatorial Theory, Ser. B* 38 (1985), pp. 73–88.

[5] N. Alon and J.H. Spencer, *The Probabilistic Method*, John Wiley & Sons, Inc., 1992.

[6] A.E. Andreev, A.E.F. Clementi, J.D.P. Rolin and L. Trevisan, "Weak Random Sources, Hitting Sets, and BPP Simulation", manuscript, February 1997.

[7] M. Bellare, O. Goldreich, and S. Goldwasser "Randomness in Interactive Proofs", *Computational Complexity*, Vol. 4, No. 4 (1993), pp. 319–354. Extended abstract in *31st FOCS*, 1990, pp. 318–326.

[8] M. Bellare, O. Goldreich, and S. Goldwasser. Addendum to [7], available from `http://theory.lcs.mit.edu/~oded/papers.html`, May 1997.

[9] M. Bellare, and J. Rompel, "Randomness-efficient oblivious sampling", *35th FOCS*, 1994.

[10] R. Canetti, G. Even and O. Goldreich, "Lower Bounds for Sampling Algorithms for Estimating the Average", *IPL*, Vol. 53, pp. 17–25, 1995.

[11] L. Carter and M. Wegman, "Universal Classes of Hash Functions", *J. Computer and System Sciences*, Vol. 18, pp. 143–154 (1979).

[12] B. Chor and O. Goldreich, "Unbiased Bits from Sources of Weak Randomness and Probabilistic Communication Complexity", *SIAM J. Comput.*, Vol. 17, No. 2, April 1988, pp. 230–261.

[13] B. Chor and O. Goldreich, "On the Power of Two–Point Based Sampling," *Jour. of Complexity*, Vol 5, 1989, pp. 96–106.

[14] A. Cohen and A. Wigderson, "Dispensers, Deterministic Amplification, and Weak Random Sources", *30th FOCS*, 1989, pp. 14–19.

[15] O. Gaber and Z. Galil, "Explicit Constructions of Linear Size Superconcentrators", *JCSS*, **22** (1981), pp. 407-420.

[16] O. Goldreich, R. Impagliazzo, L.A. Levin, R. Venkatesan, and D. Zuckerman, "Security Preserving Amplification of Hardness", *31st FOCS*, pp. 318–326, 1990.

[17] O. Goldreich and A. Wigderson, "Tiny Families of Functions with Random Properties: A Quality–Size Trade–off for Hashing", to appear in *Journal of Random structures and Algorithms.* Preliminary version in *26th STOC*, pp. 574–583, 1994.

[18] R. Impagliazzo and D. Zuckerman, "How to Recycle Random Bits", *30th FOCS*, 1989, pp. 248-253.

[19] N. Kahale, "Eigenvalues and Expansion of Regular Graphs", *Journal of the ACM*, 42(5):1091–1106, September 1995.

[20] R.M. Karp, N. Pippinger and M. Sipser, "A Time-Randomness Tradeoff", *AMS Conference on Probabilistic Computational Complexity*, Durham, New Hampshire (1985).

[21] A. Lubotzky, R. Phillips, P. Sarnak, "Explicit Expanders and the Ramanujan Conjectures", *Proc. 18th STOC*, 1986, pp. 240-246.

[22] G.A. Margulis, "Explicit Construction of Concentrators", *Prob. Per. Infor.* 9 (4) (1973), 71–80. (In Russian, English translation in *Problems of Infor. Trans.* (1975), 325–332.)

[23] N. Nisan, "Extracting randomness: how and why (a survey)". *Proceedings of the 11th Annual IEEE conference on Computational Complexity* (formerly known as Structures), IEEE 1996.

[24] M. Sipser, "Expanders, Randomness or Time vs Space", *Structure in Complexity Theory* (proceedings), 1986.

[25] L. Trevisan, "When Hamming meets Euclid: The Approximability of Geometric TSP and MST", *29th STOC*, pp. 21–29, 1997.

[26] U. Vazirani and V. Vazirani, "Random Polynomial Time Equal to Semi-Random Polynomial Time", *Proc. 26th FOCS*, pp. 417–428, 1985.

[27] D. Zuckerman, "Randomness-Optimal Sampling, Extractors, and Constructive Leader Election", *28th STOC*, 1996, pp. 286–295.

Appendix A: Expanders and Random Walks

A.1 Expanders

An (N, d, λ)-**expander** is a d-regular graph with N vertices so that the absolute value of all eigenvalues (except the biggest one) of its adjacency matrix is bounded by λ. A (d, λ)-**family** is an infinite sequence of graphs so that the n^{th} graph is a $(2^n, d, \lambda)$-expander. We say that such a family is *efficiently constructible* if there exists a log-space algorithm which given a vertex, v, in the expander and an index $i \in [d] \stackrel{\text{def}}{=} \{1, ..., d\}$, returns the i^{th} neighbor of v. We first recall that for $d = 16$ and some $\lambda < 16$, efficiently constructible $(16, \lambda)$-families do exist (cf., [15])[3].

In our applications we use (parameterized) expanders satisfying $\frac{\lambda}{d} < \alpha$ and $d = \text{poly}(1/\alpha)$, where α is an application-specific parameter. Such (parameterized) expanders are also efficiently constructible. For example, we may obtain them by taking paths of length $O(\log(1/\alpha)$ on an expander as above. Specifically, given a parameter $\alpha > 0$, we obtain an efficiently constructible (D, Λ)-family satisfying $\frac{\Lambda}{D} < \alpha$ and $D = \text{poly}(1/\alpha)$ as follows. We start with a constructible $(16, \lambda)$-family, set $k \stackrel{\text{def}}{=} \log_{16/\lambda}(1/\alpha) = O(\log 1/\alpha)$ and consider the paths of length k in each graph. This yields a constructible $(16^k, \lambda^k)$-family, and both $\frac{\lambda^k}{16^k} < \alpha$ and $16^k = \text{poly}(1/\alpha)$ indeed hold.

Comment: To obtain the best constants in Sections 4 and 5, one may use efficiently constructible Ramanujan Graphs [21]. Furthermore, using Ramanujan Graphs is essential for our proof of the second item of Theorem 4.1 as well as of Lemma 5.3. Ramanujan Graphs satisfy $\lambda \leq 2\sqrt{d}$ and so, setting $d = 4/\alpha$, we obtain $\frac{\lambda}{d} < \alpha$, where α is an application-specific parameter. Here some minor technicalities arise as these graphs are given only for certain degrees and certain sizes. Specifically, they can be efficiently constructed for $\frac{1}{2} \cdot q^k \cdot (q^{2k} - 1)$ vertices, where $q \equiv d - 1 \equiv 1 \bmod 4$ and $d - 1$ being a quadratic residue modulo q (cf., [3, Sec. II]). This technical difficulties may be resolved in two ways:

1. Fixing d and ϵ, N, we may find a q satisfying the above conditions with $\frac{1}{2} \cdot q^k \cdot (q^{2k} - 1) \in [(1-\epsilon) \cdot N, N]$, in time polynomial in $1/\epsilon$. This defines a Ramanujan Graph which is adequate for all our applications (since it biases the desired sample in $[N]$ only by ϵ).
2. Fixing d and ϵ, N, we may find a q satisfying the above conditions with $\frac{1}{2} \cdot q^k \cdot (q^{2k} - 1) \in [N, 2N]$, in time polynomial in $\log N$. We may easily modify our applications so that whenever we obtain a vertex not in $[N]$ we just ignore it. One can easily verify that the analysis of the application remains valid.

A.2 The Expander Mixing Lemma

The following lemma is folklore and has appeared in many papers. Loosely speaking, the lemma asserts that expander graphs (for which $d \gg \lambda$) have the property that the fraction of edges between two large sets of vertices approximately equals the product of the densities of these sets. This property is called *mixing*.

Lemma A.3 (Expander Mixing Lemma): *Let $G = (V, E)$ be an expander graph of degree d and λ be an upper bound on the absolute value of all eigenvalues, save the biggest one, of the adjacency*

[3] The are minor technicalities which can be easily fixed. Firstly, the Gaber–Galil expanders are defined (only) for graph sizes which are perfect squares [15]. This suffices for even n's. For odd n's, we may use a trivial modification, such as taking two copies of the graph of size 2^{n-1} and connecting each pair of corresponding vertices. Finally, we add multiple edges so that the degree becomes 16, rather than being 14 for even n's and 15 for odd n's.

matrix of the graph. Then for every two subsets, $A, B \subseteq V$*, it holds*

$$\left|\frac{|(A \times B) \cap E|}{|E|} - \frac{|A|}{|V|} \cdot \frac{|B|}{|V|}\right| \leq \frac{\lambda\sqrt{|A| \cdot |B|}}{d \cdot |V|} < \frac{\lambda}{d}$$

The lemma (and a proof) appears as Corollary 2.5 in [5, Chap. 9].

A.3 Random walks on Expanders

A fundamental discovery of Ajtai, Komlos, and Szemerédi [1] is that random walks on expander graphs provide a good approximation to repeated independent attempts to hit any arbitrary fixed subset of sufficient density (within the vertex set). The importance of this discovery stems from the fact that a random walk on an expander can be generated using much fewer random coins than required for generating independent samples in the vertex set. Precise formulations of the above discovery were given in [1, 14, 16] culminating in Kahale's optimal analysis [19, Sec. 6].

Theorem A.4 (Expander Random Walk Theorem [19, Cor. 6.1]): *Let* $G = (V, E)$ *be an expander graph of degree* d *and* λ *be an upper bound on the absolute value of all eigenvalues, save the biggest one, of the adjacency matrix of the graph. Let* W *be a subset of* V *and* $\rho \stackrel{\rm def}{=} |W|/|V|$*. Then the fraction of random walks* (in G) *of* (edge) *length* ℓ *which stay within* W *is at most*

$$\rho \cdot \left(\rho + (1-\rho) \cdot \frac{\lambda}{d}\right)^{\ell}$$

A more general bound (which is weaker for the above special case) was pointed out to us by Nabil Kahale (personal communication, April 1997):

Theorem A.5 (Expander Random Walk Theorem – general case): *Let* $G = (V, E)$*,* d *and* λ *be as above. Let* $W_0, W_1, ..., W_\ell$ *be subsets of* V *with densities* $\rho_0, ..., \rho_\ell$*, respectively. Then the fraction of random walks* (in G) *of* (edge) *length* ℓ *which intersect* $W_0 \times W_1 \times \cdots \times W_\ell$ *is at most*

$$\sqrt{\rho_0\rho_\ell} \cdot \prod_{i=1}^{\ell} \sqrt{\rho_i + (1-\rho_i) \cdot \left(\frac{\lambda}{d}\right)^2}$$

The above improves over a previous bound of [7] (see [8]). Comments regarding the proofs of both theorems follow.

Comments on the proof of Theorems A.4 and A.5

Let A be a matrix representing the random walk on G (i.e., A is the adjacency matrix of G divided by the degree, d). Let $\bar{\lambda}$ denote the absolute value of the second largest eigenvalue of A (i.e., $\bar{\lambda} \stackrel{\rm def}{=} \lambda/d$).

Let $\|x\|$ denote the Euclidean norm of $x \in \mathcal{R}^n$. For any stochastic matrix M, we let $\|M\|$ denote the norm of M; that is the maximum of $\|Mx\|$ taken over all normal vectors x (i.e., $x \in \mathcal{R}^n$ with $\|x\| = 1$). The following technical lemma is the key ingredient in both proofs.

Lemma A.6 ([19, Lem. 3.2] restated): *Let* M *be a symmetric stochastic matrix and let* δ *denote the absolute value of the second largest eigenvalue of* M*. Let* P *be a 0-1 matrix which has 1's only on the diagonal and let* ρ *be the fraction of 1's on the diagonal. Then* $\|PMP\| \leq \rho + (1-\rho) \cdot \delta$*.*

Proof of Theorem A.4: Let $u \in \mathcal{R}^n$ be the vector representing the uniform distribution (i.e., $u = (n^{-1}, ..., n^{-1})$). Let P be a 0-1 matrix so that the only 1-entries are in entries (i, i) with $i \in W$. Thus, the probability that a random walk of length ℓ stays within W is the sum of the entries of the vector

$$x \stackrel{\text{def}}{=} (PA)^\ell Pu \tag{7}$$

In other words, denoting by $\|x\|_1$ the L_1 norm of x, we are interested in an upper bound on $\|x\|_1$. Since x has at most ρn non-zero entries (i.e., $x = Px'$ for some x'), we have $\|x\|_1 \le \sqrt{\rho n} \cdot \|x\|$. Invoking Lemma A.6 we get

$$\begin{aligned}
\|x\|_1 &\le \sqrt{\rho n} \cdot \|(PA)^\ell Pu\| \\
&\le \sqrt{\rho n} \cdot \|PAP\|^\ell \cdot \|Pu\| \\
&\le \sqrt{\rho n} \cdot (\rho + (1-\rho) \cdot \bar{\lambda})^\ell \cdot \sqrt{\rho / n}
\end{aligned}$$

and the theorem follows. ■

Proof of Theorem A.5: Using the same argument, we need to upper bound the L_1 norm of x given by

$$x \stackrel{\text{def}}{=} P_\ell A \cdots P_1 A P_0 u \tag{8}$$

We observe that $\|P_j A\| = \sqrt{\|P_j A^2 P_j\|}$ and use Lemma A.6 to obtain $\|P_j A^2 P_j\| \le \rho_j + (1-\rho_j) \cdot \bar{\lambda}^2$. Thus, we have

$$\begin{aligned}
\|x\|_1 &\le \sqrt{\rho_\ell n} \cdot \|P_\ell A \cdots P_1 A P_0 u\| \\
&\le \sqrt{\rho_\ell n} \cdot \prod_{j=1}^{\ell} \|P_j A\| \cdot \|P_0 u\| \\
&\le \sqrt{\rho_\ell n} \cdot \prod_{j=1}^{\ell} \sqrt{\rho_j + (1-\rho_j) \cdot \bar{\lambda}^2} \cdot \sqrt{\rho_0 / n}
\end{aligned}$$

and the theorem follows. ■

Appendix B: Analyzing the Toeplitz Matrix Construction

For every $i \ne j$ and $a, b \in \mathrm{GF}(2)^n$, we have

$$\begin{aligned}
\Pr_{T,u}\begin{pmatrix} e_i = a \\ e_j = b \end{pmatrix} &= \Pr_{T,u}(e_i = a | e_i \oplus e_j = a \oplus b) \cdot \Pr_{T,u}(e_i \oplus e_j = a \oplus b) \\
&= \Pr_{T,u}(Tv_i + u = a | Tw = c) \cdot \Pr_T(Tw = c)
\end{aligned}$$

where $w = v_i \oplus v_j \ne 0^n$ and $c = a \oplus b$. Clearly, for any $c \in \mathrm{GF}(2)^n$ and any T':

$$\begin{aligned}
\Pr_{T,u}(Tv_i + u = a | Tw = c) &= \Pr_u(T'v_i + u = a) \\
&= 2^{-n}
\end{aligned}$$

It is thus left to show that, for any $w \ne 0^n$, when T is a uniformly chosen Toeplitz matrix, the vector Tw is uniformly distributed over $\mathrm{GF}(2)^n$. It may help to consider first the distribution of Mw, where M is a uniformly distributed n-by-n matrix. In this case Mw is merely the sum of several (not zero) uniformly and independently chosen column vectors, and so is uniformly distributed over $\mathrm{GF}(2)^n$. The argument regarding a uniformly chosen Toeplitz matrix is slightly more involved.

Let f be the first non-zero entry of $w = (w_1, ..., w_n) \neq 0^n$ (i.e., $w_1 = \cdots = w_{f-1} = 0$ and $w_f = 1$). We make the mental experiment of selecting $T = (t_{i,j})$, by uniformly selecting elements determining T as follows. First we uniformly and independently select $t_{1,n}, ..., t_{1,f}$. Next, we select $t_{2,f}, ..., t_{n,f}$ (here it is important to select $t_{j,f}$ before $t_{j+1,f}$). Finally, we select $t_{n,f-1}, ..., t_{n,1}$. Clearly, this determines a uniformly chosen Toeplitz matrix, denoted T. We conclude by showing that each of the bits of Tw is uniformly distributed given the previous bits. To prove the claim for the j^{th} bit of Tw, consider the time by which $t_{1,n}, ..., t_{1,f}, ..., t_{j-1,f}$ were determined. Note that these determine the first $j-1$ bits of Tw. The key observation is that the value of the j^{th} bit of Tw is a linear combination of the above determined values XORed with the still undetermined $t_{j,f}$. (Here we use the hypothesis that $w_1 = \cdots = w_{f-1} = 0$ and $w_f = 1$.) Thus, uniformly selecting $t_{j,f}$ makes the j^{th} bit of Tw be uniformly distributed given the past. ■

Appendix C: The Hitting problem

The hitting problem is a one-sided version of the Boolean sampling problem. Given parameters n (length), ϵ (density) and δ (error), and oracle access to any function $\sigma : \{0,1\}^n \mapsto \{0,1\}$ so that $|\{x : f(x)=1\}| \geq \epsilon 2^n$, the task to find a string which is mapped to 1. That is

Definition C.1 (hitter): *A* hitter *is a randomized algorithm that on input parameters n, ϵ and δ, and oracle access to* any *function $\sigma : \{0,1\}^n \mapsto \{0,1\}$, so that $|\{x : f(x)=1\}| \geq \epsilon 2^n$, satisfies*

$$\Pr[\sigma(\text{hitter}^{\sigma}(n, \epsilon, \delta)) = 1] \; > \; 1 - \delta$$

All results and techniques regarding sampling presented above, have simpler analogous with respect to the hitting problem. In fact this appendix may be read as a warm-up towards the entire paper.

C.1 The Information Theoretic Perspective

Analogously to the Naive Sampler, we have the Naive Hitter which *independently* selects $m \stackrel{\text{def}}{=} \frac{\ln(1/\delta)}{\epsilon}$ uniformly distributed sample points and queries the oracle on each. Clearly, the probability that the hitter fails to sample a point of value 1 is bounded above by $(1-\epsilon)^m = \delta$. The complexities of this hitter are as follows

- Sample Complexity: $m \stackrel{\text{def}}{=} \frac{\ln(1/\delta)}{\epsilon} = \Theta(\frac{log(1/\delta)}{\epsilon})$.
- Randomness Complexity: $m \cdot n = \Theta(\frac{log(1/\delta)}{\epsilon} \cdot n)$.
- Computational Complexity: indeed efficient.

It is easy to prove that the Naive Hitter is sample-optimal. That is,

Theorem C.2 *Any hitter has sample complexity bounded below by*

$$\min\left\{2^{n-O(1)}, \frac{\ln(1/2\delta)}{2\epsilon}\right\}$$

provided $\epsilon \leq \frac{1}{8}$.

Proof Sketch: Let A be a hitter with sample complexity $m = m(n, \epsilon, \delta)$ and let σ be a function selected at random by setting its value independently on each argument so that $\Pr(\sigma(x)=1) = 1.5\epsilon$. Then,

$$\Pr_\sigma[\sigma(A^\sigma(n,\epsilon,\delta)) \neq 1] \;=\; (1-1.5\epsilon)^m$$

where the probability is taken over the choice of σ and the internal coin tosses of A. On the other hand, using a Multiplicative Chernoff Bound:

$$\Pr_\sigma[|\{x : \sigma(x)=1\}| < \epsilon 2^n] = 2\exp(-\Omega(\epsilon 2^n))$$

We may assume that $\Omega(\epsilon 2^n) > \log_2(1/\delta)$ and so the probability that σ has at least ϵ fraction of 1's and yet algorithm A fails is at least $(1-1.5\epsilon)^m - \delta > \delta$, unless $m > \frac{\ln(1/2\delta)}{\ln(1-1.5\epsilon)} > \frac{\ln(1/2\delta)}{2\epsilon}$. ■

Theorem C.3 *Let $s : \mathcal{N} \times [0,1]^2 \mapsto \mathcal{R}$. Any sampler which has sample complexity bounded above by $s(n,\epsilon,\delta)$, has randomness complexity bounded below by*

$$r > n - \log_2 s(n,\epsilon,\delta) + \log_2((1-\epsilon)/\delta)$$

Proof Sketch: Let A be a hitter with sample complexity $s = s(n,\epsilon,\delta)$, and randomness complexity $r = r(n,\epsilon,\delta)$. Consider any subset of $\delta 2^r$ possible sequence of coin tosses for A and all $\delta 2^r \cdot s$ points queries at any of these coin-sequences. We argue that $\delta 2^r \cdot s > (1-\epsilon)2^n$, or else we may have a function σ be 0 on each of these points and 1 otherwise (contradicting the requirement that this function be "hit" with probability at least $1-\delta$). Thus, $r > n + \log_2(1-\epsilon) - \log_2 s + \log_2(1/\delta)$ ■

C.2 The Pairwise-Independent Hitter

Using a pairwise-independent sequence of uniformly distributed sample points rather than a totally independent one, we obtain the **pairwise-independent hitter**. Here we set $m \stackrel{\text{def}}{=} \frac{1-\epsilon}{\delta\epsilon}$. Letting ζ_i represent the σ-value of the i^{th} sample point, considering only σ's with ϵ-fraction of 1-values, and using Chebishev's Inequality we have

$$\begin{aligned}
\Pr\left(\sum_{i=1}^{m} \zeta_i = 0\right) &\leq \Pr\left(\left|m\epsilon - \sum_{i=1}^{m} \zeta_i\right| \geq \epsilon m\right) \\
&\leq \frac{m \cdot \epsilon(1-\epsilon)}{(\epsilon m)^2} \\
&= \delta
\end{aligned}$$

Recalling that we can generate $2^n - 1$ pairwise-independent samples using $2n$ coins the pairwise-independent hitter achieves

- Sample Complexity: $\frac{1}{\delta\epsilon}$ (reasonable for constant δ).
- Randomness Complexity: $2n$
- Computational Complexity: indeed efficient.

C.3 The combined Hitter

Our goal here is to decrease the sample complexity of the Pairwise-Independent Hitter while essentially maintaining its randomness complexity. To motivate the new construction we first consider an oversimplified version of it.

Combined Hitter (oversimplified): On input parameters n, ϵ and δ, set $m \stackrel{\rm def}{=} \frac{2}{\epsilon}$ and $\ell \stackrel{\rm def}{=} \log_2(1/\delta)$, generate ℓ independent m-element sequences, each being a sequence of m *pairwise-independently and uniformly distributed* strings in $\{0,1\}^n$. Denote the sample points in the $i^{\rm th}$ sequence by $s_1^i, ..., s_m^i$. We merely try all these $\ell \cdot m$ samples as hitting points. Clearly, for each $i = 1, ..., \ell$,

$$\Pr(\forall j = 1, .., m : \sigma(s_j^i) = 0) < \frac{1}{2}$$

and so the probability that none of these s_j^i "hits σ" is at most $0.5^\ell = \delta$. Thus, the oversimplified version described above is indeed a hitter and has the following complexities

- Sample Complexity: $\ell \cdot m = O(\frac{\log(1/\delta)}{\epsilon^2})$.
- Randomness Complexity: $\ell \cdot O(n) = O(n \cdot \log(1/\delta))$.
- Computational Complexity: indeed efficient.

Thus, the sample complexity is optimal (upto a constant factor), but the randomness complexity is higher than what we aim for. To reduce the randomness complexity, we use the same approach as above, but take dependent (pairwise independent) sequences rather than independent ones. The dependency we use is such which essentially preserves the probabilistic behavior of independent choices. Specifically, we use random walks on expander graphs (cf., Appendix A) to generate a sequence of ℓ "seeds" each of length $O(n)$. Each seed is used to generate a sequence of m pairwise independent elements in $\{0,1\}^n$, as above. Thus, we obtain

Corollary C.4 (The Combined Hitter): *There exists an efficient hitter with*

- Sample Complexity: $O(\frac{\log(1/\delta)}{\epsilon})$.
- Randomness Complexity: $2n + O(\log(1/\delta))$.

Furthermore, we can obtain randomness complexity $2n + (2 + o(1)) \cdot \log_2(1/\delta))$.

Proof Sketch: We use an explicit construction of expander graphs with vertex set $\{0,1\}^{2n}$, degree d and second eigenvalue λ so that $\lambda/d < 0.1$. We consider a random walk of (edge) length $\ell - 1 = \log_2(1/\delta)$ on this expander, and use each of the ℓ vertices along the path as random coins for the Pairwise-Independent Hitter which makes $m \stackrel{\rm def}{=} \epsilon/3$ trials. To analyze the performance of the resulting algorithm, we let W denote the set of coin tosses (for the basic hitter) which make the basic hitter output a point which evaluates to 1. By the hypothesis, $\frac{|W|}{2^{2n}} \leq 1/3$, and using Theorem A.4, the probability that all vertices of a random path reside in W is bounded above by $(0.34 + 0.1)^\ell < \delta$. The furthermore clause follows by using a Ramanujan Graph and an argument as in the proof of Item 2 of Theorem 4.1. ∎

C.4 The Expander Hitter

Use a Ramanujan Graph of degree $d = O(1/\epsilon\delta)$ and vertex-set $\{0,1\}^n$. Uniformly select a vertex in the graph and use its neighbors as a sample. Suppose we try to hit a 1-value of a function σ and let $S \stackrel{\rm def}{=} \{u : \sigma(u) = 1\}$. Let $B \stackrel{\rm def}{=} \{v : \mathcal{N}(v) \cap S = \emptyset\}$ be the set of bad vertices (i.e., choosing any of these results in not finding a preimage of 1). Using the Expander Mixing Lemma we have

$$\begin{aligned} \rho(B)\rho(S) &= \left| \frac{|(B \times S) \cap E|}{|E|} - \rho(B)\rho(S) \right| \\ &\leq \frac{\lambda}{d} \cdot \sqrt{\rho(B)\rho(S)} \end{aligned}$$

Hence, $\rho(B)\rho(S) \leq (\lambda/d)^2 = \epsilon\delta$ and using $\rho(S) \geq \epsilon$ we get $\rho(B) \leq \delta$. The complexities of this hitter are as follows

- Sample Complexity: $O(\frac{1}{\delta\epsilon})$ (reasonable for constant δ).
- Randomness Complexity: n
- Computational Complexity: indeed efficient.

Adapting the argument in the proof of Corollary C.4, we obtain

Corollary C.5 (The Combined Hitter, revisited): *There exists an efficient hitter with*

- Sample Complexity: $O(\frac{\log(1/\delta)}{\epsilon})$.
- Randomness Complexity: $n + (2 + o(1)) \cdot \log_2(1/\delta))$.