Extracting all the Randomness and Reducing the Error in Trevisan's Extractors*

Ran Raz† Omer Reingold‡ Salil Vadhan§

July 3, 2001

Abstract

We give explicit constructions of extractors which work for a source of any min-entropy on strings of length n. These extractors can extract any constant fraction of the min-entropy using $O(\log^2 n)$ additional random bits, and can extract all the min-entropy using $O(\log^3 n)$ additional random bits. Both of these constructions use fewer truly random bits than any previous construction which works for all min-entropies and extracts a constant fraction of the min-entropy. We then improve our second construction and show that we can reduce the entropy loss to $2\log(1/\varepsilon) + O(1)$ bits, while still using $O(\log^3 n)$ truly random bits (where entropy loss is defined as [(source min-entropy) + (# truly random bits used) − (# output bits)], and ε is the statistical difference from uniform achieved). This entropy loss is optimal up to a constant additive term.

Our extractors are obtained by observing that a weaker notion of "combinatorial design" suffices for the Nisan–Wigderson pseudorandom generator, which underlies the recent extractor of Trevisan. We give near-optimal constructions of such "weak designs" which achieve much better parameters than possible with the notion of designs used by Nisan–Wigderson and Trevisan.

We also show how to improve our constructions (and Trevisan's construction) when the required statistical difference ε from the uniform distribution is relatively small. This improvement is obtained by using multilinear error-correcting codes over finite fields, rather than the arbitrary error-correcting codes used by Trevisan.

Keywords: Extractors, Combinatorial Designs, Expander Graphs, Probabilistic Method, Pseudorandom Generators

*Preliminary versions of this work appeared in *STOC '99* [RRV99b] and on ECCC [RRV99c].

†Department of Applied Mathematics and Computer Science, Weizmann Institute, Rehovot, 76100 Israel. E-mail: ranraz@wisdom.weizmann.ac.il Work supported by an American-Israeli BSF grant 95-00238 and by ESPRIT working group RAND2.

‡AT&T Labs — Research. Building 103, 180 Park Avenue Florham Park, NJ, 07932, USA E-mail: omer@research.att.com Research performed while still at the Weizmann Institute, Rehovot, Israel. Research supported by a Clore Scholars award and an Eshkol Fellowship of the Israeli Ministry of Science and by ESPRIT working group RAND2.

§Division of Engineering and Applied Sciences. Harvard University. 33 Oxford Street. Cambridge, MA 02138. USA. E-mail: salil@eecs.harvard.edu. URL: http://www.eecs.harvard.edu/~salil. This work was done when the author was at MIT, supported by a DOD/NDSEG fellowship and partially by DARPA grant DABT63-96-C-0018.

1 Introduction

Roughly speaking, an extractor is a function which extracts (almost) truly random bits from a weak random source, using a small number of additional random bits as a catalyst. A large body of work has focused on giving explicit constructions of extractors, as such constructions have a wide variety of applications. A recent breakthrough was made by Luca Trevisan [Tre99], who discovered that the Nisan–Wigderson pseudorandom generator [NW94], previously only used in a computational setting, could be used to construct extractors. For certain settings of the parameters, Trevisan's extractor is optimal and improves on previous constructions. More explicitly, Trevisan's extractor improves over previous constructions in the case of extracting a relatively small number of random bits (e.g., extracting $k^{1-\alpha}$ bits from source with "k bits of randomness", where $\alpha > 0$ is an arbitrarily small constant) with a relatively large statistical difference from uniform distribution (e.g., constant ε, where ε is the statistical difference from uniform distribution required from the output). However, when one wants to extract more than a small fraction of the randomness from the weak random source, or when one wants to achieve a small statistical difference from uniform distribution, Trevisan's extractor performs poorly (in that a large number of truly random "catalyst" bits are needed).

In this paper, we show that Trevisan's ideas can be used in a more general and efficient way. We present two new ideas that improve Trevisan's construction. The first idea allows one to extract more than a small fraction of the randomness from the weakly random source. In particular, the idea can be used to extract all of the randomness from the weak random source. This is accomplished by replacing the "combinatorial designs" underlying the Nisan–Wigderson generator and Trevisan's construction with a weaker (and more suitable) notion. Applying a result of Wigderson and Zuckerman [WZ99] to these extractors, we also obtain improved constructions of highly expanding graphs and superconcentrators.

The second idea improves Trevisan's construction in the case where the output bits are required to be of a relatively small statistical difference from uniform distribution. The two ideas can be combined, and the final outcome is a set of new extractors that use fewer truly random bits than any previous construction which extracts at least a constant fraction of the randomness from any weak random source.

Extractors

The definition of an extractor requires quantifying two notions: how much "randomness" is in a probability distribution, and what it means for two distributions to be "close". The first is measured using a variant of entropy. A distribution X on $\{0,1\}^n$ is said to have *min-entropy* k if for all $x \in \{0,1\}^n$, $\Pr[X = x] \leq 2^{-k}$. This should be thought of as saying that X has (at least) "k bits of randomness." For example, if X is uniformly distributed on a set of size 2^k, then X has min-entropy k.

The distance measure between probability distributions used is a standard one. Two distributions X and Y on a set S are said to have *statistical difference* (or *variation distance*) ε if

$$\max_D |\Pr[D(X) = 1] - \Pr[D(Y) = 1]| = \varepsilon,$$

where the maximum is taken over all functions ("distinguishers") $D : S \to \{0,1\}$.

A function $\textsc{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is called a (k,ε)-*extractor* if for every distribution X on $\{0,1\}^n$ of min-entropy k, the induced distribution $\textsc{Ext}(X, U_d)$ on $\{0,1\}^m$ has statistical difference at most ε from U_m (where U_j denotes the uniform distribution on $\{0,1\}^j$). In other words, $\textsc{Ext}$ extracts m (almost) truly random bits from a source with k bits of hidden randomness using d additional random bits as a catalyst. The goal is to construct extractors which minimize d while m is as close to k as possible. Nonconstructively, it can be shown that for every n, $k \leq n$, and $\varepsilon > 0$, there exists a (k,ε)-extractor $\textsc{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ with $m = k$ and $d = O(\log(n/\varepsilon))$, *i.e.* all the randomness of the source is extracted using only logarithmically many additional truly random bits.[1] However, we are interested in *explicit* constructions. More precisely, a family of extractors $\{\textsc{Ext}_i : \{0,1\}^{n_i} \times \{0,1\}^{d_i} \to \{0,1\}^{m_i}\}_{i \in I}$ is called *explicit* if $\textsc{Ext}_i$ can be evaluated in time $\mathrm{poly}(n_i, d_i)$.

Dispersers are the analogue of extractors for one-sided error; instead of inducing the uniform distribution, they simply hit all but a ε fraction of points in $\{0,1\}^m$ with nonzero probability.

[1] Actually, since the extractor is fed d truly random bits in addition to the k bits of hidden randomness, one can hope to have m be close to $k + d$. This will be discussed in more detail under the heading "Entropy loss and strong extractors."

Other notations. "log" indicates the logarithm base 2 and "ln" denotes the natural logarithm. If X is a probability distribution on a finite set, we write $x \leftarrow X$ to indicate that x is selected according to X.

Previous work

Dispersers were first defined by Sipser [Sip88] and extractors were first defined by Nisan and Zuckerman [NZ96]. Much of the motivation for research on extractors comes from work done on "somewhat random sources" [SV86, CG88, Vaz87b, VV85, Vaz84, Vaz87a, CW89, Zuc96]. There have been a number of papers giving explicit constructions of dispersers and extractors, with a steady improvement in the parameters [Zuc96, NZ96, WZ99, GW97, SZ98, SSZ98, NT98, Zuc97, Ta-98, Tre99]. Most of the work on extractors was based on techniques such as k-wise independence, the Leftover hash lemma [ILL89], and various forms of composition. A new approach to constructing extractors was recently initiated by Trevisan [Tre99], who discovered a fascinating connection between constructing extractors and constructing pseudorandom generators from hard functions [Tre99]. In addition to establishing this connection, Trevisan used it to give a strikingly simple extractor construction based on the Nisan–Wigderson pseudorandom generator. This is the starting point for our work.

Explicit constructions of extractors and dispersers have a wide variety of applications, including simulating randomized algorithms with weak random sources [Zuc96]; constructing oblivious samplers [Zuc97]; constructive leader election [Zuc97, RZ98]; randomness-efficient error reduction in randomized algorithms and interactive proofs [Zuc97]; explicit constructions of expander graphs, superconcentrators, and sorting networks [WZ99]; hardness of approximation [Zuc96, Uma99]; pseudorandom generators for space-bounded computation [NZ96, RR99]; derandomizing BPP under circuit complexity assumptions [ACR97, STV99]; and other problems in complexity theory [Sip88, GZ97].

For a detailed survey of previous work on extractors and their applications, see [NT98].

Main results

The first family of extractors constructed in this paper is given in the following theorem:

Theorem 1 *For every $n, k, m \in \mathbb{N}$ and $\varepsilon > 0$, such that $m \le k \le n$, there are explicit (k, ε)-extractors* $\textsc{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *with*

1. $d = O\left(\frac{\log^2(n/\varepsilon)}{\log(k/m)}\right)$, *or*
2. $d = O\left(\log^2(n/\varepsilon) \cdot \log(1/\gamma)\right)$, *where* $1 + \gamma = k/(m-1)$, *and* $\gamma < 1/2$.

In particular, using the second extractor with $k = m$, we can extract all of the min-entropy of the source using

$$O\left(\log^2(n/\varepsilon) \cdot \log k\right)$$

additional random bits. (If ε is constant then this is just $O(\log^2 n \cdot \log k)$ additional random bits). Using the first extractor with k/m constant, we can extract any constant fraction of the min-entropy of the source using

$$O\left(\log^2(n/\varepsilon)\right)$$

additional random bits. (If ε is constant then this is just $O(\log^2 n)$ additional random bits).

An undesirable feature of the extractors in Theorem 1 (and the extractor of Trevisan [Tre99]) is that the number of truly random bits depends quadratically on $\log(1/\varepsilon)$. In (nonconstructive) optimal extractors and even some previous constructions (discussed later), this dependence is linear. Indeed, some applications of extractors, such as [RR99], require a linear dependence. In our second theorem, we improve our extractors to have a linear dependence on $\log(1/\varepsilon)$.

Theorem 2 *For every n, k, m, and ε, such that $m \le k \le n$, there are explicit (k, ε)-extractors* $\textsc{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *with*

1. $d = O\left(\frac{\log^2 n \cdot \log(1/\varepsilon)}{\log(k/m)}\right)$, *or*

2. $d = O\left(\log^2 n \cdot \log(1/\gamma) \cdot \log(1/\varepsilon)\right)$*, where* $1+\gamma = k/(m-1)$*, and* $\gamma < 1/2$*.*

Thus, in all cases, the $\log^2(n/\varepsilon)$ in Theorem 1 has been replaced with $\log^2 n \cdot \log(1/\varepsilon)$, which is an improvement when ε is relatively small. One case of note is when we want to extract $m = k^{1-\alpha}$ bits from a source of min-entropy $k \geq n^\alpha$, for an arbitrarily small constant $\alpha > 0$. This is the case in which Trevisan's extractor performs best, using $d = O(\log^2(n/\varepsilon)/\log n)$ truly random bits (which is $O(\log n)$ for $\varepsilon \geq 1/\text{poly}(n)$). In this case, Theorem 2 gives

$$d = O\left(\log n \cdot \log(1/\varepsilon)\right),$$

which is an improvement for small ε. We only provide a sketch of Theorem 2, because the results have been superseded by our recent work [RRV99a] which gives a general method to reduce the error of any extractor.

A summary of our results is given in Figure 1, and a comparison with the best previous constructions is given in Figure 2. Trevisan's construction [Tre99] uses only $O(\log^2(n/\varepsilon)/\log k)$ truly random bits but

reference	min-entropy k	output length m	additional randomness d	type
Thm. 1	any k	$m = (1-\alpha)k$	$d = O(\log^2(n/\varepsilon))$	extractor
Thm. 1	any k	$m = k$	$d = O(\log^2(n/\varepsilon) \cdot \log k)$	extractor
Thm. 2	any k	$m = k^{1-\alpha}$	$d = O(\log^2 n \cdot \log(1/\varepsilon)/\log k)$	extractor
Thm. 2	any k	$m = (1-\alpha)k$	$d = O(\log^2 n \cdot \log(1/\varepsilon))$	extractor
Thm. 2	any k	$m = k$	$d = O(\log^2 n \cdot \log(1/\varepsilon) \cdot \log k)$	extractor

Above, α is an arbitrarily small constant.

Figure 1: Summary of our constructions

reference	min-entropy k	output length m	additional randomness d	type
[GW97]	any k	$m = k$	$d = O(n - k + \log(1/\varepsilon))$	extractor
[Zuc97]	$k = \Omega(n)$	$m = (1-\alpha)k$	$d = O(\log(n/\varepsilon))$	extractor
[NT98]	any k	$m = k$	$d = O(\log^9 n \cdot \log(1/\varepsilon))$	extractor
[Ta-98]	any k	$m = k - \text{polylog}(n)$	$d = O(\log(n/\varepsilon))$	disperser
[Tre99]	any k	$m = k^{1-\alpha}$	$d = O(\log^2(n/\varepsilon)/\log k)$	extractor
ultimate goal	any k	$m = k$	$d = O(\log(n/\varepsilon))$	extractor

Above, α is an arbitrarily small constant.

Figure 2: Best previous constructions

extracts only a small fraction ($k^{1-\alpha}$) of the source min-entropy. The best previous construction that extracts all of the source min-entropy was given by Ta-Shma [NT98] and used $O(\log^9 n \cdot \log(1/\varepsilon))$ truly random bits.[2] Our extractors use more truly random bits than the extractor of [Zuc97] and the disperser of [Ta-98], but our extractors have the advantage that they work for any min-entropy (unlike [Zuc97]) and are extractors rather than dispersers (unlike [Ta-98]). The disadvantage of the extractors of [GW97] described in Figure 2 is that they only use a small number of truly random bits when the source min-entropy k is very close to the input length n (e.g., $k = n - \text{polylog}(n)$). There are also extractors given in [GW97, SZ98] which extract all of the min-entropy, but these use a small number of truly random bits only when the source min-entropy is very small (e.g., $k = \text{polylog}(n)$), and these extractors are further discussed in the context of entropy loss.

Plugging the second extractor of Theorem 1 into a construction of [WZ99] (see also [NT98]) immediately yields the following construction of highly expanding graphs:

[2]In [NT98], the number of truly random bits used by the extractor is given as $d = \text{polylog}\, n$, a polynomial of unspecified degree in $\log n$. Ta-Shma [TS98] estimates the degree of this polynomial to be 9.

Corollary 3 *For every N and $K \le N$, there is an explicitly constructible*[3] *graph on N nodes with degree $(N/K) \cdot 2^{O((\log\log N)^2(\log\log K))}$ such that every two disjoint sets of vertices of size at least K have an edge between them.*

This compares with a degree bound of $(N/K) \cdot 2^{O((\log\log N)^9)}$ due to Ta-Shma [NT98]. We also obtain similarly improved constructions of depth-2 superconcentrators, using general techniques for building them from extractors [WZ99, NT98]. These highly expanding graphs and depth-2 superconcentrators have further applications to sorting and selecting in rounds, constructing small-depth linear-sized superconcentrators, and constructing non-blocking networks [Pip87, AKSS89, WZ99], so our results translate similar improvements in each of these applications. We remark that the construction of [WZ99] used to obtain Corollary 3 requires extractors that extract nearly all the entropy of the source.

Entropy loss and strong extractors

Since a (k, ε)-extractor $\textsc{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is given k bits of hidden randomness in its first input and d truly random bits in its second input, one can actually hope for the output length m to be almost $k + d$, rather than just k. The quantity $\Delta = k + d - m$ is therefore called the *entropy loss* of the extractor. Hence, in this language, the goal in constructing extractors is to simultaneously minimize both d and the entropy loss.

Actually, in some applications of extractors, it is important not only to retain the *randomness* of the d truly random bits invested, but to explicitly retain their *values* in the output. This leads to a more stringent notion of extractors. A function $\textsc{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is called a *strong (k, ε)-extractor* if for every distribution X on $\{0,1\}^n$ of min-entropy k, the induced distribution $(U_d, \textsc{Ext}(X, U_d))$ on $\{0,1\}^d \times \{0,1\}^m$ has statistical difference at most ε from $U_d \times U_m$. Naturally, the entropy loss of a strong extractor is defined to be $\Delta = k - m$.

Nonconstructively, one can show that, for any n and $k \le n$, there exist strong extractors $\textsc{Ext}_{n,k} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{k-\Delta}$ with $d = \log(n-k) + 2\log(1/\varepsilon) + O(1)$ and entropy loss $\Delta = 2\log(1/\varepsilon) + O(1)$, and these bounds on d and Δ are tight up to additive constants (even for non-strong extractors) [RT97]. The explicit constructions, however, are still far from achieving these parameters. As for previous results, every entry in Figure 2 yields a (not necessarily strong[4]) extractor with an entropy loss of $k + d - m$, by definition. For example, the extractor of [NT98] and the disperser of [Ta-98] have entropy losses of $\mathrm{polylog}\, n$. The extractor of [GW97] is actually better than Figure 2 indicates; it is a strong extractor with an entropy loss of $n - k + O(\log(1/\varepsilon))$ (though this is only interesting when k is very close to n). In addition, the "tiny families of hash functions" of [SZ98] give strong extractors with $d = O(k + \log n)$ and entropy loss $2\log(1/\varepsilon) + O(1)$; these have optimal entropy loss but are only interesting when k is very small (e.g., $k = \mathrm{polylog}\, n$), as d is linear in k. (The fact that d here does not explicitly depend on ε is not a contradiction to the lower bound on d, as no nontrivial extraction is occurring when $k < \Delta$ and the lower bounds do not apply.)

Our extractors are in fact strong extractors. Moreover, by combining the second extractors of Theorem 1 and Theorem 2 with the low min-entropy extractors of [SZ98], we are able to achieve optimal entropy loss (up to an additive constant):

Theorem 4 *For every $n, k \in \mathbb{N}$, and $\varepsilon > 0$ such that $k \le n$, there are explicit strong (k, ε)-extractors $\textsc{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{k-\Delta}$ with entropy loss*

$$\Delta = 2\log(1/\varepsilon) + O(1),$$

and

1. $d = O\left(\log^2(n/\varepsilon) \cdot \log k\right)$, *or*
2. $d = O\left(\log^2 n \cdot \log(1/\varepsilon) \cdot \log k\right)$.

[3] By explicitly constructible, we mean that, given N and K, the graph can be constructed deterministically in time $\mathrm{poly}(N)$.

[4] However, subsequent to this work, it was shown how to transform any extractor into a strong extractor, with only a small cost in the other parameters [RSW00].

In particular, in order for the output of the extractor to have statistical difference .01 from uniform, one need only lose a constant number of bits of entropy. A comparison of this result with previous results on entropy loss is given in Figure 3.

reference	additional randomness d	entropy loss Δ	type	strong?
[GW97]	$d = O(n - k + \log(1/\varepsilon))$	$\Delta = n - k + 12\log(1/\varepsilon) + O(1)$	extractor	yes
[GW97]	$d = n - k + 2\log(1/\varepsilon) + O(1)$	$\Delta = 2\log(1/\varepsilon) + O(1)$	extractor	no
[SZ98]	$d = O(k + \log n)$	$\Delta = 2\log(1/\varepsilon) + O(1)$	extractor	yes
[NT98]	$d = O(\log^9 n \cdot \log(1/\varepsilon))$	$\Delta = O(\log^9 n \cdot \log(1/\varepsilon))$	extractor	no
[Ta-98]	$d = O(\log(n/\varepsilon))$	$\Delta = \mathrm{polylog}(n/\varepsilon)$	disperser	no
Thm. 4	$d = O\left(\log^2(n/\varepsilon) \cdot \log k\right)$	$\Delta = 2\log(1/\varepsilon) + O(1)$	extractor	yes
Thm. 4	$d = O\left(\log^2 n \cdot \log(1/\varepsilon) \cdot \log k\right)$	$\Delta = 2\log(1/\varepsilon) + O(1)$	extractor	yes
nonconstructive & optimal [RT97]	$d = \log(n - k) + 2\log(1/\varepsilon) + O(1)$	$\Delta = 2\log(1/\varepsilon) + O(1)$	extractor	yes

All of the above work for any source of min-entropy k.

Figure 3: Results on entropy loss

Techniques and Tools

Here we briefly highlight some of the techniques and tools used to achieve our improvements. In particular, we give the definitions of both combinatorial designs and weak designs.

Designs. The main combinatorial objects underlying the Nisan–Wigderson pseudorandom generator and subsequently Trevisan's extractors are collections of sets with small pairwise intersections. Following [NW94], we will refer to these as *designs*, but in the combinatorics literature, they are often called *packings* (cf., [AS00, Sec. 4.7]).

Definition 5 *For $\ell \in \mathbb{N}$ and $\rho \geq 1$, a family of sets $S_1, \ldots, S_m \subset [d]$ is an (ℓ, ρ)-*design *if*

1. *For all i, $|S_i| = \ell$.*
2. *For all $i \neq j$, $|S_i \cap S_j| \leq \log \rho$.*

The first improvement of this paper stems from the observation that actually a weaker form of designs suffices for the analysis of [NW94, Tre99]. As it turns out, it is sufficient to use a set system in which the quantity $\max_i \sum_{j<i} 2^{|S_i \cap S_j|}$ is small (in contrast to designs, in which $\max_{i \neq j} |S_i \cap S_j|$ is bounded). We call such set systems *weak designs*.

Definition 6 *A family of sets $S_1, \ldots, S_m \subset [d]$ is a* weak (ℓ, ρ)-design *if*

1. *For all i, $|S_i| = \ell$.*
2. *For all i,*

$$\sum_{j<i} 2^{|S_i \cap S_j|} \leq \rho \cdot (m - 1).$$

Clearly, every (ℓ, ρ)-design is also an (ℓ, ρ)-weak design, but not conversely. Indeed, we will see that, for some settings of parameters, it is possible to have the universe size d much smaller for weak designs than the corresponding designs. This will allow our extractors to attain a much smaller seed length than Trevisan's extractors when extracting most or all of the min-entropy from the source. The weak designs used in the first extractor of Theorem 1 are constructed using an application of the Probabilistic Method, which we then derandomize using the Method of Conditional Expectations (cf., [AS00] and [MR95, Ch. 5]). We then apply a simple iteration to these first weak designs to obtain the weak designs used in the second extractor. We also prove a lower bound showing that our weak designs are near-optimal.

Multilinear error-correcting codes. The second improvement of the paper (reducing the dependence of the seed length on ε) is achieved by using a specific error-correcting code rather than an arbitrary one. More specifically, we use multilinear error-correcting codes over finite fields. The main property we use is that the restriction of a multilinear function to a subset of its input variables is still a multilinear function. We can hence bound the description size of that restriction by the description size of a multilinear function rather than the description size of an arbitrary function. This turns out to be very useful in the extractor analysis.

A general method to obtain optimal entropy loss. For our third improvement, we observe that a method of Wigderson and Zuckerman [WZ99] (with a slightly refined analysis) along with the "low min-entropy" extractor of Srinivasan and Zuckerman [SZ98] can be used to reduce the entropy loss of *any extractor* to the optimal value of $2\log 1/\varepsilon + O(1)$. This transformation increases the seed length of an extractor by at most $O(\log n + \Delta)$ bits, where Δ is the initial entropy loss. Applying this general transformation to the extractors of Theorem 1 and Theorem 2 gives the extractors of Theorem 4.

Organization

In Section 2, we analyze the Trevisan extractor and show that weak designs can be used instead of standard designs in this construction. Section 3 contains our results on the construction of weak designs. We also give lower bounds showing that the parameters achieved by our weak designs are impossible for standard designs, and that our constructions of them are nearly optimal. Theorem 1 is proven in Section 4 (as a corollary of the results of Sections 2 and 3). In Section 5, we sketch our method for improving the dependence on the error as claimed in Theorem 2. In Section 6, we give a general method for obtaining optimal entropy loss in extractors. In Section 7, we show that using a relaxed notion of designs also gives some quantitative improvements over [NW94] in the construction of pseudorandom generators from hard Boolean functions. We conclude with a discussion of subsequent work and open problems in Section 8.

2 The extractor

In this section, we describe the Trevisan extractor and present a more refined analysis of it. Most importantly, we show that weak designs can be used instead of standard designs. We also show that it is in fact a strong extractor (i.e., the seed can be given as part of the output). Aside from these two improvements, the description of the extractor follows [Tre99] very closely. The main tool in the Trevisan extractor is the Nisan–Wigderson pseudorandom generator [NW94]. Let $\mathcal{S} = (S_1, \ldots, S_m)$ be a collection of subsets of $[d]$ of size ℓ, and let $P : \{0,1\}^\ell \to \{0,1\}$ be any Boolean function. For a string $y \in \{0,1\}^d$, define $y|_{S_i}$ to be the string in $\{0,1\}^\ell$ obtained by projecting y onto the coordinates specified by S_i. Then the Nisan–Wigderson generator $\mathrm{NW}_{\mathcal{S},P}$ is defined as

$$\mathrm{NW}_{\mathcal{S},P}(y) = P(y|_{S_1}) \cdots P(y|_{S_m}).$$

In addition to the Nisan–Wigderson generator, the Trevisan extractor makes use of error-correcting codes. We need codes satisfying the following lemma. Such codes can be obtained using standard techniques; for completeness a proof is given in Appendix A.

Lemma 7 (error-correcting codes) *For every $n \in \mathbb{N}$ and $\delta > 0$ there is a code $\mathrm{EC}_{n,\delta} : \{0,1\}^n \to \{0,1\}^{\overline{n}}$ where $\overline{n} = \mathrm{poly}(n, 1/\delta)$ such that every Hamming ball of relative radius $1/2 - \delta$ in $\{0,1\}^{\overline{n}}$ contains at most $1/\delta^2$ codewords. Furthermore, $\mathrm{EC}_{n,\delta}$ can be evaluated in time $\mathrm{poly}(n, 1/\delta)$ and $\overline{n}$ can be assumed to be a power of 2.*

We can now describe the Trevisan extractor, which takes as parameters $n, m, k \in \mathbb{N}$, and $\varepsilon > 0$, where $m \le k \le n$. Let $\mathrm{EC} : \{0,1\}^n \to \{0,1\}^{\overline{n}}$ be as in Lemma 7, with $\delta = \varepsilon/4m$ and define $\ell = \log \overline{n} = O(\log n/\varepsilon)$. For $u \in \{0,1\}^n$, we view $\mathrm{EC}(u)$ as a Boolean function $\overline{u} : \{0,1\}^\ell \to \{0,1\}$. Let $\mathcal{S} = (S_1, \ldots, S_m)$ be a collection of subsets of $[d]$ (for some d) such that $|S_i| = \ell$ for each i. (How $\mathcal{S}$ is selected will crucially affect the performance of the extractor; we will later choose it to be one of our weak designs.)

Then the extractor $\mathrm{Ext}_{\mathcal{S}} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is defined as

$$\mathrm{Ext}_{\mathcal{S}}(u, y) = \mathrm{NW}_{\mathcal{S},\overline{u}}(y) = \overline{u}(y|_{S_1}) \cdots \overline{u}(y|_{S_m}).$$

We will now analyze this extractor. The following lemma, due to Yao, allows us to focus on "next-bit predictors" instead of distinguishers.

Lemma 8 ([Yao82]) *Suppose that $\langle Y, Z\rangle$ is a distribution on $\{0,1\}^d \times \{0,1\}^m$ s.t. Y is uniformly distributed over $\{0,1\}^d$ yet the statistical difference of $\langle Y, Z\rangle$ from $U_d \times U_m$ is greater than ε. Then there is an $i \in [m]$ and a function ("next-bit predicator") $A : \{0,1\}^d \times \{0,1\}^{i-1} \to \{0,1\}$ such that*

$$\Pr_{\langle y,z\rangle \leftarrow \langle Y,Z\rangle} [A(y, z_1 z_2 \cdots z_{i-1}) = z_i] > \frac{1}{2} + \frac{\varepsilon}{m}.$$

Moreover, if there is a circuit $D : \{0,1\}^d \times \{0,1\}^m \to \{0,1\}$ of size s distinguishing $\langle Y, Z\rangle$ from $U_d \times U_m$ with advantage ε, then A may also be taken to be of circuit complexity s.

We will not use the "moreover" part of Lemma 8 in the analysis of our extractor; it will only be used for our quantitative improvement to the pseudorandom generators of [NW94] given in Section 7.

The following lemma is a refinement of ones in [NW94, Tre99]. It shows how, from any next-bit predictor A for $\mathrm{NW}_{\mathcal{S},P}$, one can obtain a "program" of small description size (or circuit complexity) which, using A as an oracle, computes P with noticeable advantage.

Lemma 9 *Fix $\mathcal{S}$. For every $i \in [m]$, there is a set $\mathcal{F}_i$ of functions from $\{0,1\}^\ell$ to $\{0,1\}^{d+i-1}$ (depending only on $\mathcal{S}$ and i) such that*

1. *For every function $P : \{0,1\}^\ell \to \{0,1\}$ and every predictor $A : \{0,1\}^{d+i-1} \to \{0,1\}$, there exists a function $f \in \mathcal{F}_i$ such that*

$$\Pr_x [A(f(x)) = P(x)] \ge \Pr_y \left[A\left(y, P(y|_{S_1}) \cdots P(y|_{S_{i-1}})\right) = P(y|_{S_i})\right],$$

where x is selected uniformly from $\{0,1\}^\ell$ and y from $\{0,1\}^d$.

2. $\log |\mathcal{F}_i| \le d + \sum_{j<i} 2^{|S_i \cap S_j|}$.
3. *Each function in $\mathcal{F}_i$ can be computed by a circuit of size $O\left(\sum_{j<i} \left(2^{|S_i \cap S_j|} - 1\right)\right)$.*[5]

The main improvement over [NW94, Tre99] in Lemma 9 is the use of $\sum_{j<i} 2^{|S_i \cap S_j|}$ rather than $(i-1) \cdot 2^{\max_j |S_i \cap S_j|}$ in the bound on $|\mathcal{F}_i|$. This refined bound illustrates the connection with weak designs. We will not use Item 3 (the bound on circuit size) in the analysis of our extractor; we only use this for the construction of pseudorandom generators in Section 7.

Proof: Let

$$\alpha = \Pr_y \left[A\left(y, P(y|_{S_1}) \cdots P(y|_{S_{i-1}})\right) = P(y|_{S_i})\right]$$

By an averaging argument we can fix all the bits of y outside S_i while preserving the prediction probability. Renaming $y|_{S_i}$ as x, we now observe that x varies uniformly over $\{0,1\}^\ell$ while $P(y|_{S_j})$ for $j \ne i$ is now a function P_j of x that depends on only $|S_i \cap S_j|$ bits of x. So, we have

$$\Pr_x [A(y(x), P_1(x) \cdots P_{i-1}(x)) = P(x)] \ge \alpha.$$

[5]We measure circuit size by the number of *internal* gates, so, for example, the identity function has circuit size 0.

Therefore, it suffices to let $\mathcal{F}_i$ be the set of functions f of the form $x \mapsto (y(x), P_1(x), P_2(x), \ldots, P_{i-1}(x))$, where $P_j(x)$ depends only some set T_{ij} of bits of x, where $|T_{ij}| = |S_i \cap S_j|$. The number of bits it takes to represent each P_j is $2^{|T_{ij}|} = 2^{|S_i \cap S_j|}$. Also, $y(x)$ is simply a function that places x in the positions indexed by S_i and is fixed in the other $d - \ell$ positions. So, the total number of bits it takes to represent a function in $\mathcal{F}_i$ is at most $d - \ell + \sum_{j<i} 2^{|S_i \cap S_j|}$, giving the desired bound on $\log |\mathcal{F}_i|$. For the bound on circuit size, notice that the circuit size of f is simply the sum of the circuit sizes of the P_j's, and every function on t bits can be computed by a circuit of size $c \cdot (2^t - 1)$, for some constant c (cf., [Weg87, Thm. 2.2]). Note that this bound on circuit size is even true for $t = 0$, since we count the circuit size of a constant function as 0. ■

We now analyze the extractor $\mathrm{Ext}_{\mathcal{S}}$ when we take $\mathcal{S}$ to be a weak design. The argument follows the analysis of Trevisan's extractor in [Tre99] except that we use the more refined bounds on $|\mathcal{F}_i|$ given by Lemma 9.

Proposition 10 *If $\mathcal{S} = (S_1, \ldots, S_m)$ (with $S_i \subset [d]$) is a weak (ℓ, ρ)-design for $\rho = (k - 3\log(m/\varepsilon) - d - 3)/m$, then $\mathrm{Ext}_{\mathcal{S}} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is a strong (k, ε)-extractor.*

Proof: Let X be any distribution of min-entropy k. We need to show that the statistical difference between $\langle U_d \, \mathrm{Ext}(X, U_d)\rangle$ and $U_d \times U_m$ is at most ε. By Lemma 8, it suffices to show that for every next-bit predictor $A\colon \{0,1\}^d \times \{0,1\}^{i-1} \to \{0,1\}$,

$$\Pr_{u \leftarrow X, y}\left[A\left(y, \overline{u}(y|_{S_1}) \cdots \overline{u}(y|_{S_{i-1}})\right) = \overline{u}(y|_{S_i})\right] \le \frac{1}{2} + \frac{\varepsilon}{m}$$

where y is selected uniformly from $\{0,1\}^d$. So let $A\colon \{0,1\}^d \times \{0,1\}^{i-1} \to \{0,1\}$ be any next-bit predictor and let $\mathcal{F}_i$ be as in Lemma 9, so that $|\mathcal{F}_i| < 2^{d+\rho m}$.

Let B be the set of u for which there exists an $f \in \mathcal{F}_i$ such that $\Pr_x[A(f(x)) = \overline{u}(x)] > 1/2 + \varepsilon/2m$. In other words, B is the set of "bad" u for which $\overline{u}$ can be approximated by a function of small "description size" relative to A. Now a counting argument will show that this can only happen with small probability, since $\overline{u}$ is a codeword in an error-correcting code selected according to a distribution with high min-entropy. By the property of the error-correcting code given in Lemma 7, for each function $f \in \mathcal{F}_i$, there are at most $(2m/\varepsilon)^2$ strings $u \in \{0,1\}^n$ such that $\Pr_x[A(f(x)) = \overline{u}(x)] > 1/2 + \varepsilon/2m$. By the union bound,

$$|B| \le (2m/\varepsilon)^2 \cdot |\mathcal{F}_i| < (2m/\varepsilon)^2 \cdot 2^{d+\rho m}.$$

Since X has min-entropy k, each $u \in B$ has probability at most 2^{-k} of being selected from X, so

$$\begin{aligned}
\Pr_{u \leftarrow X}[u \in B] &< \left((2m/\varepsilon)^2 2^{d+\rho m}\right) \cdot 2^{-k} \\
&= \left((2m/\varepsilon)^2 2^{d+k-3\log(m/\varepsilon)-d-3}\right) \cdot 2^{-k} \\
&= \varepsilon/2m
\end{aligned}$$

Now, by Lemma 9, if $u \notin B$, then

$$\Pr_{y}\left[A\left(y, \overline{u}(y|_{S_1}) \cdots \overline{u}(y|_{S_{i-1}})\right) = \overline{u}(y|_{S_i})\right] \le \frac{1}{2} + \frac{\varepsilon}{2m}.$$

Thus,

$$\begin{aligned}
\Pr_{u \leftarrow X, y}\left[A\left(y, \overline{u}(y|_{S_1}) \cdots \overline{u}(y|_{S_{i-1}})\right) = \overline{u}(y|_{S_i})\right] &\le \Pr_{u \leftarrow X}[u \in B] + \Pr_{u \leftarrow X}[u \notin B] \cdot \left(\frac{1}{2} + \frac{\varepsilon}{2m}\right) \\
&\le \frac{\varepsilon}{2m} + \left(\frac{1}{2} + \frac{\varepsilon}{2m}\right) \\
&= \frac{1}{2} + \frac{\varepsilon}{m}.
\end{aligned}$$

■

The bound $\rho = (k - 3\log(m/\varepsilon) - d - 3)/m$ can be viewed as follows: we start with min-entropy k, then we immediately incur a (typically small) entropy loss of $3\log(m/\varepsilon) - d - 3$, and then the output m is a $1/\rho$ fraction of what is left.

Remark 11 Two improvements to the above analysis can slightly reduce the entropy loss of $3\log(m/\varepsilon) + d + O(1)$ incurred in Proposition 10. We only give brief sketches of these improvements, since the savings will be completely subsumed by our general entropy loss reduction technique in Section 6. First, d bits of entropy loss can be completely eliminated by doing things a little differently. Specifically, the bits of y outside S_i can be fixed at the same time as when Lemma 8 is applied and absorbed into the predictor A. The key point is that these bits need not depend on the particular sample u selected from the source X and hence they need not count towards the "description size" of u.

Second, the $3\log(m/\varepsilon)$ bits of entropy loss can be improved to $2\log(1/\varepsilon)+3\log m+O(1)$. This is achieved by partitioning the set of "bad" u for which $\overline{u}$ can be approximated by a function of the form $A(f(\cdot))$, into sets B_j according to the quality of approximation (e.g., take B_j to be those u for which $\overline{u}$ can be approximated with error between $1/2 - 2^j\varepsilon/2m$ and $1/2 - 2^{j-1}\varepsilon/2m$). Then we use an error-correcting code in which for every $\delta' \geq \delta$ (rather than just $\delta' = \delta$) any Hamming ball of relative radius $1/2 - \delta'$ contains at most $1/(\delta')^2$ codewords. Doing the analysis separately for each B_j has the effect of balancing the probability that $u \leftarrow X$ lands in B_j against the maximum advantage possible for u in B_j.

Remark 12 An interesting feature of our extractors is that for $m' \leq m$, the m'-bit prefix of the output is essentially the same as if the extractor had been constructed for output length m'. This makes it possible to construct (k, ε)-extractors $\textsc{Ext}$ with the property that even if $\textsc{Ext}$ is applied to a source of (unknown) min-entropy $k' < k$, an $m' = m'(k')$ prefix of the output will still be ε-close to uniform. To implement this idea, one should use our construction of weak designs in Lemma 15, which has the property that the weak (ℓ, ρ)-design $(S_1, ..., S_m)$ constructed is such that for every i, $(S_1, ..., S_i)$ is also a weak (ℓ, ρ)-design.

3 Designs — constructions and lower bounds

To motivate our design constructions, observe that in extractor analysis above, the parameters of a design correspond to the parameters of the extractor as follows (in the discussion below the parameter ε of the extractor is fixed, for simplicity, to be some small constant):

$$\begin{aligned} \text{source min-entropy} &\approx \rho m \\ \text{output length} &= m \\ \text{input length} &= 2^{\Theta(\ell)} \\ \text{additional randomness} &= d \end{aligned}$$

Hence, our goal in constructing designs is to minimize d given parameters m, ℓ, and ρ (such that $\rho \geq 1$). Notice that $1/\rho$ is essentially the fraction of the source min-entropy that is extracted, so ideally ρ would be as close to 1 as possible.

The construction of designs used in Trevisan's extractor is given by the following lemma (rediscovered in [NW94, Tre99]):

Lemma 13 ([EFF85]) *For $m, \ell, d \in \mathbb{N}$ and $\rho > 1$, there exists an efficiently constructible (ℓ, ρ)-design $S_1, \ldots, S_m \subseteq [d]$ if $m \leq \binom{d}{\lfloor \log 2\rho \rfloor} / \binom{\ell}{\lfloor \log 2\rho \rfloor}^2$. In particular, for any $m, \ell \in \mathbb{N}$ and $\rho > 1$, one exists with*

$$d = O\left(\frac{\ell^2 \cdot m^{1/\log\rho}}{\log\rho}\right).$$

Notice that the dependence of d on ρ is very poor. In particular, if we want to extract a constant fraction of the min-entropy, we need more than $m^{\Omega(1)}$ truly random bits. This is unavoidable with the current definition of designs: if $\rho < 2$, then all the sets must be disjoint, so $d \geq m\ell$. In general, we have the following lower bound, which is well-known in the "packing" literature (cf., [Röd85]).

Proposition 14 *If $S_1, \ldots, S_m \subset [d]$ is an (ℓ, ρ)-design, then $m \leq \binom{d}{\lfloor \log 2\rho \rfloor} / \binom{\ell}{\lfloor \log 2\rho \rfloor}$. In particular,*

$$d \geq m^{1/\log 2\rho} \cdot (\ell - \log \rho)$$

Proof: Let $I = \lfloor \log 2\rho \rfloor > \max_{i \neq j} |S_i \cap S_j|$. For each $j = 1, \ldots, m$, let Γ_j be the set of subsets of S_j of size I, so $|\Gamma_j| = \binom{\ell}{I}$. Let $\Gamma = \bigcup_j \Gamma_j$. Notice that the sets Γ_j are disjoint, because no two distinct sets S_i, S_j share I or more elements. Thus, $|\Gamma| = m \cdot \binom{\ell}{I}$. At the same time, $|\Gamma|$ consists of subsets of $[d]$ of size I, so $|\Gamma| \leq \binom{d}{I}$. So we have

$$m \cdot \binom{\ell}{I} \leq \binom{d}{I},$$

establishing the first bound of the proposition. The "in particular" part is obtained as follows.

$$m \leq \frac{\binom{d}{I+1}}{\binom{\ell}{I+1}} = \left(\frac{d}{\ell}\right)\left(\frac{d-1}{\ell-1}\right)\cdots\left(\frac{d-I}{\ell-I}\right) \leq \left(\frac{d}{\ell-I}\right)^{I+1}.$$

■

We will now show that, for many settings of m, ℓ, and ρ, there exist weak (ℓ, ρ)-designs $S_1, \ldots, S_m \subset [d]$ with much smaller values of d than possible with (ℓ, ρ)-designs. In particular, in the following lemma, we give a construction of weak designs where the universe size d *does not depend on m,* the number of sets.

Lemma 15 *For every $\ell, m \in \mathbb{N}$ and $\rho > 1$, there exists a weak (ℓ, ρ)-design $S_1, \ldots, S_m \subset [d]$ with*

$$d = \left\lceil \frac{\ell}{\ln \rho} \right\rceil \cdot \ell.$$

Moreover, such a family can be found in time $\mathrm{poly}(m, d)$.

Proof: Let ℓ, m, and ρ be given, and let $d = \lceil \ell / \ln \rho \rceil \cdot \ell$. We view $[d]$ as the disjoint union of ℓ blocks $B_1, \ldots, B_\ell$, each of size $\lceil \ell / \ln \rho \rceil$. We construct the sets $S_1, \ldots, S_m$ in sequence so that

1. Each set contains exactly one element from each block, and
2. $\sum_{j<i} 2^{|S_i \cap S_j|} \leq \rho \cdot (i-1)$.

Existence. Suppose we have $S_1, \ldots, S_{i-1} \subset [d]$ satisfying the above conditions. We prove that there exists a set S_i satisfying the required conditions using the Probabilistic Method [AS00] (see also [MR95, Ch. 5]). Let $a_1, \ldots, a_\ell$ be uniformly and independently selected elements of $B_1, \ldots, B_\ell$, respectively, and then let $S_i = \{a_1, \ldots, a_\ell\}$. We will argue that with nonzero probability, Condition 2 holds. Let $Y_{j,k}$ be the indicator random variable for the event $a_k \in S_j$, so $\Pr[Y_{j,k} = 1] = 1/|B_k| = 1/\lceil \ell / \ln \rho \rceil$. Notice that for a fixed j, the random variables $Y_{j,1}, \ldots, Y_{j,\ell}$ are independent.

$$\begin{aligned}
\mathrm{E}\left[\sum_{j<i} 2^{|S_i \cap S_j|}\right] &= \sum_{j<i} \mathrm{E}\left[2^{\sum_k Y_{j,k}}\right] \\
&= \sum_{j<i} \mathrm{E}\left[\prod_k 2^{Y_{j,k}}\right] \\
&= \sum_{j<i} \prod_k \mathrm{E}\left[2^{Y_{j,k}}\right] \\
&= (i-1) \cdot \left(1 + \frac{1}{\lceil \ell / \ln \rho \rceil}\right)^{\ell} \\
&\leq (i-1) \cdot \rho
\end{aligned}$$

Hence, with nonzero probability, Condition 2 holds, so a set S_i satisfying the requirements exists. However, we want to find such a set deterministically. This can be accomplished by a straightforward application of the Method of Conditional Expectations (see [AS00] and [MR95, Ch. 5]), as proceed to show.

Derandomization. Above, we showed that $\mathrm{E}\left[\sum_{j<i} 2^{|S_i \cap S_j|}\right] \le \rho \cdot (i-1)$. By averaging, this implies that there exists an $\alpha_1 \in B_1$ such that

$$\mathrm{E}\left[\sum_{j<i} 2^{|S_i \cap S_j|} \,\middle|\, a_1 = \alpha_1\right] \le \rho \cdot (i-1) \tag{1}$$

So, assuming we can efficiently calculate the conditional expectation $\mathrm{E}\left[\sum_{j<i} 2^{|S_i \cap S_j|} \,\middle|\, a_1 = \alpha_1\right]$ for every $\alpha_1 \in B_1$, we can find the α_1 that makes Inequality 1 hold. Then, fixing such an α_1, another averaging argument implies that there exists an $\alpha_2 \in B_2$ such that

$$\mathrm{E}\left[\sum_{j<i} 2^{|S_i \cap S_j|} \,\middle|\, a_1 = \alpha_1, a_2 = \alpha_2\right] \le \rho \cdot (i-1) \tag{2}$$

Again, assuming that we can compute the appropriate conditional expectations, we can find an α_2 that makes Inequality 2 hold. Proceeding like this, we obtain $\alpha_1, \ldots, \alpha_\ell$ such that

$$\mathrm{E}\left[\sum_{j<i} 2^{|S_i \cap S_j|} \,\middle|\, a_1 = \alpha_1, a_2 = \alpha_2, \ldots, a_\ell = \alpha_\ell\right] \le \rho \cdot (i-1) \tag{3}$$

But now there is no more randomness left in the experiment, and Inequality 3 simply says that $\sum_{j<i} 2^{|S_i \cap S_j|} \le \rho \cdot (i-1)$, for $S_i = \{\alpha_1, \ldots, \alpha_\ell\}$. To implement this algorithm for finding S_i, we need to be able to calculate the conditional expectation

$$\mathrm{E}\left[\sum_{j<i} 2^{|S_i \cap S_j|} \,\middle|\, a_1 = \alpha_1, \ldots, a_t = \alpha_t\right],$$

for any t and $\alpha_1, \ldots, \alpha_t$. If we let $T = \{\alpha_1, \ldots, \alpha_i\}$, then a calculation like the one in the proof of Lemma 15 for the unconditional expectation shows

$$\mathrm{E}\left[\sum_{j<i} 2^{|S_i \cap S_j|} \,\middle|\, a_1 = \alpha_1, \ldots, a_t = \alpha_t\right] = \sum_{j<i} 2^{|T \cap S_j|} \cdot \left(1 + \frac{1}{\lceil \ell / \ln \rho \rceil}\right)^{\ell - t},$$

which can be easily computed. ■

Remark 16 A perhaps more natural way to carry out the above probabilistic construction is to chose S_i uniformly from the set of all subsets of $[d]$ of size ℓ, rather than dividing $[d]$ into ℓ blocks. This gives essentially the same bounds, but complicates the analysis because the elements of S_i are no longer independent.

Lemma 15 already gives something much better than Lemma 13; for constant $\rho > 1$, d is $O(\ell^2)$ instead of $\ell^2 \cdot m^{\Omega(1)}$. However, as ρ gets very close to 1, d gets very large. Specifically, if $\rho = 1 + \gamma$ for small γ, then the above gives $d = O(\ell^2/\gamma)$. To improve this, we notice that the proof of Lemma 15 does not take advantage of the fact that there are fewer terms in $\sum_{j<i} 2^{|S_i \cap S_j|}$ when i is small; indeed the proof actually shows how to obtain $\sum_{j<i} 2^{|S_i \cap S_j|} < \rho \cdot (i-1)$.[6] Since we only need a bound of $\rho \cdot (m-1)$ for all i, this suggests that we should "pack" more sets in the beginning. This packing is accomplished by iterating the construction of Lemma 15 (directly inspired by the iteration of Wigderson and Zuckerman [WZ99] on extractors), and yields the following improvement.

[6] In fact it is *necessary* that $d = \Omega(\ell^2/\log \rho)$ if $\sum_{j<i} 2^{|S_i \cap S_j|} < \rho \cdot (i-1)$ for all i. See Remark 19.

Lemma 17 *For every $\ell, m \in \mathbb{N}$ and $0 < \gamma < 1/2$, there exists a weak $(\ell, 1+\gamma)$-design $S_1, \ldots, S_m \subset [d]$ with*

$$d = O\left(\ell^2 \cdot \log \frac{1}{\gamma}\right).$$

In particular, for every $\ell, m \in \mathbb{N}$, there exists a weak $(\ell, 1)$-design $S_1, \ldots, S_m \subset [d]$ with

$$d = O\left(\ell^2 \cdot \log m\right).$$

Moreover, these families can be found in time $\mathrm{poly}(m, d)$.

Proof: Let $d_0 = \lceil \ell / \ln 2 \rceil \cdot \ell$, $h = \lceil \log(4/\gamma) \rceil$, and $d = h \cdot d_0 = O(\ell^2 \cdot \log(1/\gamma))$. We view $[d]$ as the disjoint union of h blocks $B_0, \ldots, B_{h-1}$ each of size d_0. For each $t \in \{0, \ldots, h-1\}$, let $n_t = \lfloor (1-2^{-t}) \cdot (1+\gamma/2) \cdot m \rfloor$ and $m_t = n_{t+1} - n_t$. Note that $n_h \geq m$.

Now we define our weak design $S_1, \ldots, S_m$. For each $t \in \{0, \ldots, h-1\}$, we let $S_{n_t+1}, \ldots, S_{n_t+m_t} \subset B_t$ be a weak $(\ell, 2)$-design as given by Lemma 15. In other words, we take the ordered union of h weak $(\ell, 2)$-designs (consisting of $m_1, m_2, \ldots, m_h$ sets, respectively) using disjoint subsets of the universe for each. The number of sets is $n_h \geq m$, the size of the universe is d, and each set is of size ℓ, so we only need to check that for all $i \in [m]$, $\sum_{j<i} 2^{|S_i \cap S_j|} < \rho \cdot (m-1)$. For $i \in \{n_t + 1, \ldots, n_t + m_t\}$, S_i is disjoint from any S_j for any $j \leq n_t$ and

$$\sum_{j=n_t+1}^{i-1} 2^{|S_i \cap S_j|} \leq 2 \cdot (m_t - 1)$$

since $S_{n_t+1}, \ldots, S_{n_t+m_t}$ is a weak $(\ell, 2)$-design. Thus, we have

$$\begin{aligned}
\sum_{j<i} 2^{|S_i \cap S_j|} &= \sum_{j=1}^{n_t} 2^{|S_i \cap S_j|} + \sum_{j=n_t+1}^{i-1} 2^{|S_i \cap S_j|} \\
&\leq n_t + 2 \cdot (m_t - 1) \\
&= 2 \cdot n_{t+1} - n_t - 2 \\
&< 2 \cdot (1 - 2^{-t-1}) \cdot (1+\gamma/2) \cdot m - (1 - 2^{-t}) \cdot (1+\gamma/2) \cdot m - 1 \\
&\leq (1+\gamma) \cdot (m-1),
\end{aligned}$$

as desired (except when $m = 1$, which is a trivial case). The "in particular" part of Lemma 17 follows because every weak $(\ell, 1+1/m)$-design is actually a weak $(\ell, 1)$-design (since $\lfloor (1+1/m) \cdot (m-1) \rfloor = m-1$). ■

In terms of our extractors, Lemma 17 translates to extracting essentially all of the entropy of a source on $\{0,1\}^n$ of min-entropy k using $d = O(\log^2 n \cdot \log k)$ truly random bits (as we will prove formally in Section 4). For extractors which use only $O(\log n)$ truly random bits, one would need $d = O(\ell)$. However, one cannot hope to do better than $\Omega(\ell^2)$ using the current analysis with weak designs. Indeed, the following proposition shows that our weak designs are optimal up to the $\log(1/\gamma)$ factor in our second construction.

Proposition 18 *For every weak (ℓ, ρ)-design $S_1, \ldots, S_m \subset [d]$,*

$$d \geq \Omega\left(\min\left\{\frac{\ell^2}{\log 2\rho}, m\ell\right\}\right)$$

Notice that $d = m\ell$ can be trivialily achieved having all the sets disjoint. Moreover, $\log 2\rho$ approaches 1 as ρ approaches 1, so the lower bound for $m \geq \ell$ and $\rho \approx 1$ is $\Omega(\ell^2)$.

Proof: Let $\overline{m} = \lceil 2d/\ell \rceil$. If $\overline{m} \geq m$, then $d \geq \ell \cdot (m-1)/2 = \Omega(m\ell)$ and we are done, so we may assume that $\overline{m} < m$. We will now consider only the first $\overline{m}$ sets. We have

$$\rho \geq \max_i \frac{1}{\overline{m}-1} \sum_{j<i} 2^{|S_i \cap S_j|}$$

$$
\begin{aligned}
&\geq \frac{1}{\overline{m}(\overline{m}-1)} \sum_{i=1}^{\overline{m}} \sum_{j<i} 2^{|S_i \cap S_j|} \\
&= \frac{1}{2} \left(\frac{1}{\binom{\overline{m}}{2}} \sum_{j<i\leq \overline{m}} 2^{|S_i \cap S_j|} \right) \\
&\geq \frac{1}{2} \cdot 2^{\left(\frac{1}{\binom{\overline{m}}{2}} \sum_{j<i\leq \overline{m}} |S_i \cap S_j| \right)}
\end{aligned}
$$

where the last inequality is an application of Jensen's inequality. Thus,

$$
\log 2\rho > \frac{2}{\overline{m}^2} \sum_{j<i\leq \overline{m}} |S_i \cap S_j| \tag{4}
$$

By the inclusion-exclusion bound,

$$
\begin{aligned}
\sum_{j<i\leq \overline{m}} |S_i \cap S_j| &\geq \left(\sum_{i=1}^{\overline{m}} |S_i| \right) - \left| \bigcup_{i=1}^{\overline{m}} S_i \right| \\
&\geq \overline{m}\ell - d \\
&= 2d - d = d
\end{aligned}
$$

Putting this in Inequality 4, we have

$$
\log 2\rho > \frac{2d}{\overline{m}^2} = \frac{2d}{(2d/\ell)^2} = \frac{\ell^2}{2d},
$$

which proves the proposition. ■

Remark 19 The above proof gives a stronger bound on d if we have a family of sets $S_1, \ldots, S_m$ such that for all i, $\sum_{j<i} 2^{|S_i \cap S_j|} < \rho \cdot (i-1)$ (e.g., the family of sets constructed in the proof of Lemma 15). If we have such a bound, then summing over i from 1 to $\overline{m}$ gives

$$
\rho \cdot \binom{\overline{m}}{2} = \sum_{i\leq \overline{m}} \rho \cdot (i-1) > \sum_{j<i\leq \overline{m}} 2^{|S_i \cap S_j|},
$$

and applying Jensen's inequality and taking logs as in the above proof gives

$$
\log \rho > \frac{2}{\overline{m}^2} \sum_{j<i\leq \overline{m}} |S_i \cap S_j|
$$

instead of Inequality 4. Following the rest of the proof without change, this shows that

$$
d \geq \Omega \left(\min \left\{ \frac{\ell^2}{\log \rho}, m\ell \right\} \right).
$$

Remark 20 Using an information-theoretic analogue of the inclusion-exclusion bound, due to Impagliazzo and Wigderson [IW96], one can generalize the lower bound of Proposition 18 to a wider class of generators with similar properties to the Nisan–Wigderson generator. Specifically, one can prove the following:

Proposition 21 *Suppose $X = (X_1, \ldots, X_m)$ and $Y = (Y_1, \ldots, Y_m)$ are (jointly distributed) random variables such that*

1. *For all i, $\mathrm{H}(X_i|Y_i) \geq \ell$, and*

2. For all i,

$$\sum_{j<i} 2^{\mathrm{H}(X_j|Y_i)} \leq \rho \cdot (m-1).$$

Then

$$\mathrm{H}(X) \geq \Omega\left(\min\left\{\frac{\ell^2}{\log 2\rho}, m\ell\right\}\right).$$

In Proposition 21, $\mathrm{H}(\cdot)$ denotes the entropy function and $\mathrm{H}(\cdot|\cdot)$ denotes conditional entropy (cf., [CT91]). Impagliazzo and Wigderson [IW97] had previously given a statement like Proposition 21 with the second condition replaced by $\max_{i,j} \mathrm{H}(X_j|Y_i) \leq \log \rho$; ours is a generalization to the analogue of "weak designs." The proof directly follows the proof of Proposition 18, replacing the usual inclusion-exclusion bound with that of [IW96], which states that $\mathrm{H}(X) \geq \sum_i \mathrm{H}(X_i|Y_i) - \sum_{j<i} \mathrm{H}(X_j|Y_i)$.

To compare Proposition 21 with the Nisan–Wigderson generator $\mathrm{NW}_{\mathcal{S},P}$, let $X_i = x|_{S_i}$ and $Y_i = x|_{\overline{S_i}}$, where x is chosen uniformly at random. In the analysis of our extractor, the properties of the Nisan–Wigderson generator we use are that, conditioned on $Y_i = y$, X_i takes on all possible values in $\{0,1\}^\ell$ whereas $\sum_{j<i} n_j \leq \rho \cdot (m-1)$, where n_j is the number of values that X_j can take on given that $Y_i = y$. The properties required by the hypothesis of Proposition 21 are even weaker.

4 Putting it Together

Theorem 1 almost follows immediately by combining Proposition 10 with the weak designs given by Lemmas 15 and 17. The only technicality is that Proposition 10 does not allow us to take $\rho = k/m$ (or $k/(m-1)$) which is what we would need to deduce Theorem 1 directly. Instead, we use $\rho = (k-\Delta)/m$ for some $\Delta = O(d)$ and consequently lose Δ bits of the source entropy in Proposition 10. However, since Δ will be relatively small, we can compensate for this by giving our extractor Δ more truly random bits in its seed (increasing the seed length by only a constant factor) which we just concatenate to the output to compensate for the loss. We give the details of this below, starting by presenting our extractors as *strong* extractors with an extra entropy loss of $\Delta = O(d)$.

Theorem 22 *For every $n, k, m \in \mathbb{N}$ and $\varepsilon > 0$, such that $m \leq k \leq n$, there are explicit strong (k, ε)-extractors* $\mathrm{EXT} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{m-\Delta}$ *with*

1. $d = O\left(\frac{\log^2(n/\varepsilon)}{\log(k/m)}\right)$, *or*

2. $d = O\left(\log^2(n/\varepsilon) \cdot \log(1/\gamma)\right)$, *where* $1+\gamma = k/(m-1)$, *and* $\gamma < 1/2$,

and $\Delta = O(d)$.

Proof: Let $\Delta \geq 1$ be an integer parameter to be set later, and let $k' = k - \Delta + 1$, $m' = m - \Delta$, $\rho = k/(m-1) \leq k'/m'$, and let $\ell = \Theta(\log n/\varepsilon)$ (as in our extractor construction). For Part 1 (resp., Part 2), let $\mathcal{S} = (S_1, \ldots, S_{m'})$, with $S_i \subset [d']$ be the weak (ℓ, ρ)-design of Lemma 15 (resp., Lemma 17), so $d' = O\left(\frac{\log^2(n/\varepsilon)}{\log(k/m)}\right)$ (resp., $d' = O(\log^2(n/\varepsilon)\log(1/\gamma))$). Since d' is independent of Δ, we can set $\Delta = d + 3\log(m/\varepsilon) + 4 = O(d)$. Proposition 10 tells us that $\mathrm{EXT}_{\mathcal{S}} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{m'}$ is a strong (k, ε)-extractor. ■

To obtain Theorem 1, we simply add Δ extra bits to the seed and concatenate them to the output. (Recall that Theorem 1 does not claim strong extractors).

5 Reducing the error

The construction given above works well and improves over previous constructions when ε is relatively large. However, the number d of truly random bits needed is quadratic in $\log(1/\varepsilon)$, which is not as good as the linear dependency achieved by some previous constructions. In this section, we improve this quadratic dependency in our constructions (and in Trevisan's construction) to a linear dependency. We only sketch the proof in this section, as even better extractors can be obtained using our recent work [RRV99a].

The quadratic dependence on $\log(1/\varepsilon)$ in our extractor arises from the fact that an (ℓ, ρ)-weak design requires a universe whose size grows quadratically with ℓ (cf., Proposition 18). In the extractor of the previous section (and Trevisan's extractor), ℓ is taken to be the logarithm of the length of the error-correcting code used (as we view codewords as functions $P : \{0,1\}^{\ell} \to \{0,1\}$). The analysis of the extractor reveals that in order to achieve a small statistical difference ε from uniform, we must use an error-correcting code with very good distance properties; namely, one in which no Hamming ball of radius $1/2 - O(\varepsilon/m)$ contains many codewords. However, an error-correcting code with such a strong distance property must have length at least $\mathrm{poly}(n, \varepsilon)$, resulting in $\ell = \Omega(\log(n/\varepsilon))$, and a seed length that is quadratic in $\log(1/\varepsilon)$.

The solution we give in this section is to use an error-correcting code over a large alphabet F, in which we view every codeword as a function from F^{ℓ} to F rather than a function from $\{0,1\}^{\ell}$ to $\{0,1\}$. Then it is possible to have a code with very good distance properties (relative to ε) with ℓ being independent of ε; only the alphabet size F need depend on ε. Using this approach, we encounter two problems. The first problem is that the function which computes the codeword P given a predictor A (as in Lemma 9) will be built from functions of the form $P_j : F^{|S_i \cap S_j|} \to F$. In the proof of Lemma 9, we bounded the description size of the P_j's by the description size of an arbitrary function $F^{|S_i \cap S_j|} \to F$, which is $2^{|S_i \cap S_j|}$ when $F = \{0,1\}$. But, as F increases in size, this bound on description size becomes too large to handle. The second problem is that, when we use a large alphabet, the output of the extractor consists of elements of F rather than bits. We will not be able to argue that these elements of F are uniformly distributed, but rather that (with high enough probability) the i'th element of F in the output is unpredictable given the first $i-1$ elements of F.

The solution to the first problem comes from our choice of error-correcting codes. We use multilinear error correcting codes (over finite fields) rather than the arbitrary error correcting codes used in Section 2. We can then make use of the fact that the restriction of a multilinear function to a subset of its input variables is still a multilinear function. We can hence bound the description size of that restriction by the description size of a multilinear function rather than the description size of an arbitrary function.

The second problem can be solved using standard techniques. Specifically, the fact that the i'th component of the output is (almost always) unpredictable given the first $i-1$ components means that the output is what is known as an (almost) *block-wise source* [CG88]. In our case, the block-wise source has blocks of logarithmic length, and standard techniques can be used to extract truly random bits from such a source using a small number of additional truly random bits.

Let F be some fixed finite field such that $\log|F| \approx c \cdot \log(n/\varepsilon)$, where c is some sufficiently large constant (say $c = 10$). For $\varepsilon \geq 1/n$, the dependence on ε in the extractors of Theorem 1 can be absorbed into the hidden constant. Thus, we will only need to use the constructions of this section in case $\varepsilon < 1/n$, and hence we may assume that

$$\log|F| = O(\log(1/\varepsilon)).$$

In this section, we think of an extractor $\mathrm{EXT} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ as a function

$$\mathrm{EXT} : F^{n'} \times F^{d'} \to F^{m'},$$

where $n' = n/(\log|F|)$, $d' = d/(\log|F|)$ and $m' = m/(\log|F|)$. (We assume for simplicity that n', d', m', $\log n'$, and $\log|F|$ are all integers.)

Let $\mathcal{S} = (S_1, \ldots, S_{m'})$ be a collection of subsets of $[d']$ such that $|S_i| = \ell$ for each i, and let $P : F^{\ell} \to F$ be any function. For a string $y \in F^{d'}$, define $y|_{S_i}$ to be the string in F^{ℓ} obtained by projecting y onto the coordinates specified by S_i. Then we define $\mathrm{NW}'_{\mathcal{S},P}$ as

$$\mathrm{NW}'_{\mathcal{S},P}(y) = P(y|_{S_1}) \cdots P(y|_{S_{m'}}).$$

We will use in this section $\ell = \log n'$; note that ℓ is bounded by $\log n$, independent of ε. Let G be the set of all functions from F^{ℓ} to F. A *multilinear* function from F^{ℓ} to F is a function of ℓ variables over F that is linear (over F) in each one of the variables. There are $|F|^{2^\ell} = |F|^{n'}$ multilinear functions from F^ℓ to F (one needs to specify 2^ℓ coefficients), so we may define an error-correcting code $\mathrm{EC} : F^{n'} \to G$ which associates to each element u of $F^{n'}$ a distinct multilinear function $\mathrm{EC}(u) = \overline{u} : F^\ell \to F$. The distance property of this code is formalized by the following standard bound:

Lemma 23 (cf., [GRS00, Thm. 17]) *For every function $Q : F^\ell \to F$, there are at most $O(\sqrt{|F|/\ell})$ codewords (i.e., multilinear functions) that agree with Q in at least a $\sqrt{2\ell/|F|}$ fraction of the points in F^ℓ.*

We define the function $\text{BW-E}\textsc{xt}_{\mathcal{S}} : F^{n'} \times F^{d'} \to F^{m'}$ as

$$\text{BW-E}\textsc{xt}_{\mathcal{S}}(u, y) = \mathrm{NW}'_{\mathcal{S},\overline{u}}(y) = \overline{u}(y|_{S_1}) \cdots \overline{u}(y|_{S_{m'}}).$$

(The function BW-Ext is not our final extractor. Rather, it will be used to extract an (almost) block-wise source; hence the notation BW-Ext.) Note that the number of truly random bits used by BW-Ext is $d' \log |F| = O(d' \cdot \log(1/\varepsilon))$. The following lemma is analogous to Lemma 9. It shows how, from any next-element predictor A for $\mathrm{NW}'_{\mathcal{S},P}$, one can obtain a "program" of small description size (or circuit complexity) which, using A as an oracle, computes P with noticeable advantage.

Lemma 24 *Fix $\mathcal{S}$. For every $i \in [m']$, there is a set $\mathcal{F}_i$ of functions from F^ℓ to $F^{d'+i-1}$ (depending only on $F, \mathcal{S}$ and i) such that*

1. *For every multilinear function $P : F^\ell \to F$ and every predictor $A : F^{d'+i-1} \to F$, there exists a function $f \in \mathcal{F}_i$ such that*

$$\Pr_x [A(f(x)) = P(x)] \geq \Pr_y \left[A\left(y, P(y|_{S_1}) \cdots P(y|_{S_{i-1}})\right) = P(y|_{S_i})\right],$$

   where x is selected uniformly from F^ℓ and y from $F^{d'}$.

2. $\log |\mathcal{F}_i| \leq \left(d' + \sum_{j<i} 2^{|S_i \cap S_j|}\right) \cdot \log |F|$.

For the proof, we use the fact that the restriction of a multilinear function to a subset of its input variables is a multilinear function, and the fact that the logarithm of the number of multilinear functions in $|S_i \cap S_j|$ variables is $2^{|S_i \cap S_j|} \cdot \log |F|$. Otherwise, the proof is similar to the one of Lemma 9.

Now assume that $\mathcal{S}$ is a weak (ℓ, ρ)-design for $\rho = (k - d - c \cdot \log |F|)/m$ (where, say, $c = 10$), and let X be any distribution of min-entropy k. The following proposition shows that $\text{BW-E}\textsc{xt}(X, U_d)$ doesn't have a good next-element predictor. The proposition is analogous to Proposition 10.

Proposition 25 *If $\mathcal{S} = (S_1, \ldots, S_{m'})$ (with $S_i \subset [d']$) is a weak (ℓ, ρ)-design for $\rho = (k - d - c \cdot \log |F|)/m$ (where c is some sufficiently large constant, say $c = 10$), and X is a distribution of min-entropy k then for every next-element predictor $A : F^{d'+i-1} \to F$,*

$$\Pr_{u \leftarrow X, y} \left[A\left(y, \overline{u}(y|_{S_1}) \cdots \overline{u}(y|_{S_{i-1}})\right) = \overline{u}(y|_{S_i})\right] \leq \frac{1}{|F|^\delta}$$

where δ is some (not too small) constant (say $\delta = 1/4$), and where y is selected uniformly from $F^{d'}$.

The proof is similar to the one of Proposition 10, except that we use the distance property of multilinear error-correcting codes given by Lemma 23 and we use Lemma 24 rather than Lemma 9.

In general, the function $\text{BW-E}\textsc{xt}_{\mathcal{S}}$ is not a good extractor. Nevertheless, by Proposition 25, we know that (with probability $m \cdot \left(1/|F|^{\Omega(1)}\right)$) each element $\text{BW-E}\textsc{xt}(X, U_d)$ has large min-entropy ($\Omega(\log |F|)$ bits) given all its predecessors. That is, it is close to a "block-wise source" in the sense of [CG88], in which the min-entropy of each block given the predecessors (and the seed) is a constant fraction of its length (which is $\log |F|$). We can now construct an extractor from $\text{BW-E}\textsc{xt}_{\mathcal{S}}$ in one of the following ways:

1. By applying on the entire output $\text{BW-E}\textsc{xt}(X, U_d)$ the extractor of [Zuc97] that extracts a constant fraction of the min-entropy as long as the min-entropy is at least a constant fraction of the number of bits.

2. By applying on each element of BW-$\textsc{Ext}(X, U_d)$ a pairwise independent hash function $h : \{0,1\}^{\log|F|} \to \{0,1\}^{\delta' \cdot \log|F|}$, where δ' is some small constant (we can apply the same hash function on all the elements). This is a special case of the block-wise extraction methods of [CG88, NZ96].

Both ways are very efficient in terms of the number of additional random bits needed.

The first part of Theorem 2 is now obtained by using the weak designs given by Lemma 15 (as in the proof of Theorem 1). The resulting seed length (using an (ℓ, ρ)-weak design for $\rho = (k - d - c \cdot \log|F|)/m$) is

$$d = O(d' \log(1/\varepsilon)) = O\left(\frac{\ell^2}{\log \rho} \cdot \log(1/\varepsilon)\right).$$

However, the number of bits we extract is only $\delta' \cdot \log|F| \cdot m' = \delta' m \approx \delta' k/\rho$, for some constant $\delta' < 1$. Hence, we can only directly use this to extract up to a small constant fraction of the min-entropy (even if we use the weak designs of Lemma 17). In order to extract more of the min-entropy of the source, we will need to use iterations, as in [WZ99] (cf., Lemma 26). A constant number of iterations will allow us to extract any constant fraction of the min-entropy. In general, to obtain $m = k/(1+\gamma)$, we will need $O(\log(1/\gamma))$ iterations and hence we need $O\left(\log^2 n \cdot \log(1/\varepsilon) \cdot \log(1/\gamma)\right)$ additional random bits.

6 Achieving optimal entropy loss

In this section we give a general method for reducing the entropy loss of extractors (recall that the entropy loss of an extractor is the quantity $\Delta = k + d - m$). Applied to the extractors of previous sections, this transformation gives the extractors of Theorem 4 which have optimal entropy loss (up to a constant additive term) of $2\log 1/\varepsilon + O(1)$.

We use an idea due to Wigderson and Zuckerman [WZ99]: Suppose we have a strong (k, ε)-extractor $\textsc{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{k-\Delta}$ with entropy loss Δ. Now, if X is a source of min-entropy k, then conditioned on "most" values of $(U_d, \textsc{Ext}(X, U_d))$, X will still have min-entropy close to Δ. So, we can use a different extractor (with fresh truly random bits) to extract some more of this min-entropy. This is formalized by the following lemma, which slightly strengthens one in [WZ99]:

Lemma 26 *Let $s > 0$. Suppose $\textsc{Ext}_1 : \{0,1\}^n \times \{0,1\}^{d_1} \to \{0,1\}^{m_1}$ is a strong (k, ε_1)-extractor with entropy loss Δ_1 and $\textsc{Ext}_2 : \{0,1\}^n \times \{0,1\}^{d_2} \to \{0,1\}^{m_2}$ is a strong $(\Delta_1 - s, \varepsilon_2)$-extractor with entropy loss Δ_2. Define $\textsc{Ext} : \{0,1\}^n \times \{0,1\}^{d_1+d_2} \to \{0,1\}^{m_1+m_2}$ by*

$$\textsc{Ext}(x, (y_1, y_2)) = \textsc{Ext}_1(x, y_1) \circ \textsc{Ext}_2(x, y_2),$$

where $\circ$ denotes concatenation. Then $\textsc{Ext}$ is a strong $\left(k, \left(\frac{1}{1-2^{-s}}\right) \cdot \varepsilon_1 + \varepsilon_2\right)$ with entropy loss $\Delta_2 + s$.

The main difference from the corresponding lemma in [WZ99] is that the statistical difference from uniform in $\textsc{Ext}$ has a better dependence on s (in [WZ99], the expression is $\varepsilon_1 + \varepsilon_2 + 2^{-s}$). There is also an analogue of Lemma 26 for non-strong extractors; in that case, $\textsc{Ext}_2$ should be applied to the pair (x, y_1) rather than just x. Details can be found in the preliminary version of this work [RRV99b].

Before proving Lemma 26, let us see how it can be used to obtain extractors with near optimal entropy loss. Note that the entropy loss of $\textsc{Ext}$ in Lemma 26 only depends on the entropy loss of $\textsc{Ext}_2$. Furthermore, $\textsc{Ext}_2$ needs to work for min-entropy Δ_1 (the entropy loss of $\textsc{Ext}_1$) which will be relatively small (i.e., $O(d_1)$) in our application. We therefore need extractors which work well for small min-entropy (and in particular have very small entropy loss). Such extractors are the "low min-entropy" extractors of Srinivasan and Zuckerman [SZ98]:

Lemma 27 ([SZ98]) *For every n, $k \le n$, and $\varepsilon > 0$, there is an explicit strong (k, ε)-extractor $\textsc{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{k-\Delta}$ with entropy loss $\Delta = 2\lceil \log(1/\varepsilon) \rceil + 2$ and $d = O(k + \log n)$.*

The transformation we were looking for (that reduces the entropy loss of extractors) is now obtained as a simple corollary of Lemma 26 and Lemma 27:

Lemma 28 *Let* $\textsc{Ext}_1 : \{0,1\}^n \times \{0,1\}^{d_1} \to \{0,1\}^{k-\Delta_1}$ *be any strong* $(k, \varepsilon/4)$*-extractor with entropy loss* Δ_1*. Then there exists a strong* (k,ε)*-extractor* $\textsc{Ext} : \{0,1\}^n \times \{0,1\}^{d_1+d_2} \mapsto \{0,1\}^{k-\Delta}$ *such that*

1. $\textsc{Ext}$ *has entropy loss* $\Delta = 2\lceil \log(1/\varepsilon)\rceil + 5$*,*
2. $d_2 = O(\Delta_1 + \log n)$*, and*
3. $\textsc{Ext}$ *is computable in polynomial time with one oracle query to* $\textsc{Ext}_1$*.*

Proof: By Lemma 27, there is an explicit strong $(\Delta_1 - 1, \varepsilon/2)$-extractor $\textsc{Ext}_2 \colon \{0,1\}^n \times \{0,1\}^{d_2} \to \{0,1\}^{\Delta_1 - 1 - \Delta_2}$ with $d_2 = O(\Delta_1 + \log n)$ and entropy loss $\Delta_2 = 2\lceil\log(2/\varepsilon)\rceil + 2 = 2\lceil \log(1/\varepsilon)\rceil + 4$. Combining $\textsc{Ext}_1$ and $\textsc{Ext}_2$ via Lemma 26 (with $s = 1$) gives a (k,ε)-extractor $\textsc{Ext} \colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{k-\Delta}$ such that

1. $\textsc{Ext}$ has entropy loss $\Delta = \Delta_2 + 1 = 2\lceil\log(1/\varepsilon)\rceil + 5$
2. $d = d_1 + d_2$.
3. Since $\textsc{Ext}_2$ is explicit, $\textsc{Ext}$ is computable in polynomial time with one oracle query to $\textsc{Ext}_1$.

■

The transformation of Lemma 28 applies to any extractor $\textsc{Ext}_1$.[7] However, it is especially interesting when the entropy loss of $\textsc{Ext}_1$ is $O(d_1)$. In this case, the seed length of $\textsc{Ext}$ is only larger by a constant factor than the seed length of $\textsc{Ext}_1$ (i.e., the transformation is not "too costly"). One example of such extractors is the construction of [NT98] (where the seed length and entropy loss are both $O(\log^9 n)$). Other examples are our constructions in the previous sections. Applying Lemma 28 to these extractors give the extractors of Theorem 4 as a simple corollary:

Proof of Theorem 4: Part 1 of the theorem is obtained by applying Lemma 28 to the second extractor of Theorem 22 (with $m = k$) as $\textsc{Ext}_1$. Part 2 of the theorem is obtained by applying Lemma 28 to the second extractor of (the strong extractor version of) Theorem 2. ■

Proof of Lemma 26: Let X be any source of min-entropy k. Let L be the set of pairs $(y,z) \in \{0,1\}^{d_1} \times \{0,1\}^{m_1}$ such that $\Pr[\textsc{Ext}_1(X,y) = z] < 2^{-(m_1+s)}$ (i.e., L is the set of pairs (y,z) for which z is "light" under $\textsc{Ext}_1(\cdot, y)$, in the sense that it occurs with probability that is smaller than uniform by a factor of 2^s). Recall that the entropy loss is defined as $\Delta_1 = k - m_1$.

Claim 29 *For every* $(y,z) \notin L$*, the conditional distribution of* X *given that* $\textsc{Ext}_1(X,y) = z$ *has min-entropy at least* $\Delta_1 - s$*.*

Proof of claim: For every x such that $\textsc{Ext}_1(x,y) = z$,

$$\begin{aligned} \Pr[X = x | \textsc{Ext}_1(X,y) = z] &= \frac{\Pr[X = x]}{\Pr[\textsc{Ext}_1(X,y) = z]} \\ &\le \frac{2^{-k}}{2^{-m_1 - s}} \\ &= 2^{-(\Delta_1 - s)}. \end{aligned}$$

This proves Claim 29. □

Thus, since $\textsc{Ext}_2$ is a $(\Delta_1 - s, \varepsilon_2)$-extractor, for every $(y,z) \notin L$, the conditional distribution of $(U_{d_2}, \textsc{Ext}_2(X, U_{d_2}))$ given that $(U_{d_1}, \textsc{Ext}_1(X, U_{d_1})) = (y,z)$ has statistical difference at most ε_2 from uniform. Now we argue that $(U_{d_1}, \textsc{Ext}_1(X, U_{d_1}))$ lands in L with low probability.

[7] Although the lemma is stated only for strong extractors, an analogous lemma holds for non-strong extractors (using the analogue of Lemma 26 for non-strong extractors mentioned above).

Claim 30 $\Pr[(U_{d_1}, \text{Ext}_1(X, U_{d_1})) \in L] \le \varepsilon_1/(2^s - 1)$.

> **Proof of claim:** For every $(y,z) \in L$, $\Pr[U_{d_1} \times U_{m_1} = (y,z)] > 2^s \cdot \Pr[(U_{d_1}, \text{Ext}_1(X, U_{d_1})) = (y,z)]$. Thus, $\Pr[U_{d_1} \times U_{m_1} \in L] \ge 2^s \cdot \Pr[(U_{d_1}, \text{Ext}_1(X, U_{d_1})) \in L]$. Now, by definition, the statistical difference between $U_{d_1} \times U_{m_1}$ and $(U_{d_1}, \text{Ext}_1(X, U_{d_1}))$ is at least
>
> $$\Pr[U_{d_1} \times U_{m_1} \in L] - \Pr[(U_{d_1}, \text{Ext}_1(X, U_{d_1})) \in L] \ge (2^s - 1) \cdot \Pr[(U_{d_1}, \text{Ext}_1(X, U_{d_1})) \in L].$$
>
> Since Ext_1 is a strong (k, ε_1)-extractor, this statistical difference is at most ε_1, and the claim follows. $\square$

Thus, $(U_{d_1}, \text{Ext}_1(X, U_{d_1})) \circ (U_{d_2}, \text{Ext}_2(X, U_{d_2}))$ can be described as a joint distribution $A \circ B$ with following properties:

1. A has statistical difference at most ε_1 from $U_{d_1} \times U_{m_1}$.
2. With probability at least $1 - \delta$ over $a \leftarrow A$, $B|_{A=a}$ has statistical difference at most ε_2 from $U_{d_2} \times U_{m_2}$ (where $\delta = \varepsilon_1/(2^s - 1)$).

From this, it follows that $A \circ B$ has statistical difference at most $\varepsilon_1 + \delta + \varepsilon_2 = \left(\frac{1}{1-2^{-s}}\right) \cdot \varepsilon_1 + \varepsilon_2$ from $U_{d_1} \times U_{m_1} \times U_{d_2} \times U_{m_2} = U_{d_1+d_2} \times U_{m_1+m_2}$, proving Lemma 26. $\blacksquare$

7 Better pseudorandom generators

Using alternative types of designs also gives some quantitative improvements in the construction of pseudorandom generators from hard predicates in [NW94]. From Lemma 9, we see that the relevant notion of design in the setting of pseudorandom generation versus small circuits is the following:

Definition 31 *A family of sets $S_1, \ldots, S_m \subset [d]$ is a* type 2 weak (ℓ, ρ)-design *if*

1. *For all i, $|S_i| = \ell$.*
2. *For all i,*

$$\sum_{j<i} \left(2^{|S_i \cap S_j|} - 1\right) \le \rho \cdot (m-1).$$

Notice that it is meaningful to consider even values of ρ less than 1, since $2^{|S_i \cap S_j|} - 1$ can be zero. Using the the same construction as Lemma 15, we obtain:

Lemma 32 *For every $\ell, m \in \mathbb{N}$ and $\rho > 0$, there exists a type 2 weak (ℓ, ρ)-design $S_1, \ldots, S_m \subset [d]$ with*

$$d = \left\lceil \frac{\ell}{\ln(1+\rho)} \right\rceil \cdot \ell.$$

Moreover, such a family can be found in time $\text{poly}(m, d)$.

Proof: The construction of a weak (ℓ, ρ')-design in Lemma 15 actually gives a family of sets $S_1, \ldots, S_m$ such that for all i, $\sum_{j<i} 2^{|S_i \cap S_j|} \le \rho' \cdot (i-1)$, which implies that $\sum_{j<i} \left(2^{|S_i \cap S_j|} - 1\right) \le (\rho' - 1) \cdot (i-1) \le (\rho' - 1) \cdot (m-1)$. Setting $\rho' = \rho + 1$, we have a type 2 weak (ℓ, ρ) design, and the universe size is $d = \lceil \ell / \ln \rho' \rceil \cdot \ell = \lceil \ell / \ln(1+\rho) \rceil \cdot \ell$. $\blacksquare$

The quantitative relation between pseudorandom generators and (type 2) weak designs follows readily from Lemmas 8 and 9:

Lemma 33 *Suppose $P\colon \{0,1\}^\ell \to \{0,1\}$ is a predicate such that no circuit of size s can compute P correctly on more than a fraction $\frac{1}{2}+\varepsilon$ of the inputs and suppose that $\mathcal{S} = (S_1,\ldots,S_m)$ where $S_i \subset [d]$ is a type 2 weak (ℓ,ρ)-design. Then no circuit of size $s - O(\rho \cdot m)$ can distinguish $\mathrm{NW}_{\mathcal{S},P}$ from uniform with advantage greater than $m\varepsilon$.*

Combining this and Lemma 32 with $s = 2m$ and ρ a small constant, we obtain

Theorem 34 *Suppose $P\colon \{0,1\}^\ell \to \{0,1\}$ is a predicate such that no circuit of size $2m$ can compute P correctly on more than a fraction $\frac{1}{2}+\frac{\varepsilon}{m}$ of the inputs. Then there is a generator $G_{P,m}\colon \{0,1\}^{O(\ell^2)} \to \{0,1\}^m$ computable in time $\mathrm{poly}(m,\ell)$, making m oracle calls to P, such that no circuit of size m can distinguish the output of G from uniform with advantage greater than ε.*

In other words, to obtain m bits which are pseudorandom against circuits of size m, we need only assume that there is a predicate which is hard against circuits of size $O(m)$. In contrast, the results of [NW94] always need to assume that the predicate is hard against circuits of size $m^{1+\epsilon}$ for some constant $\epsilon > 0$ (or else their generator will require a seed length that is polynomial in m instead of ℓ). In fact, if we instead take $\rho = 1/\ell$, we need only assume that the predicate is hard against circuits of size $(1+1/\ell)\cdot m$ (and the generator will have a seed length $O(\ell^3)$).

8 Subsequent Work and Open Problems

Subsequent to the original version of this work [RRV99b], there have been several papers giving further improved extractor constructions, which bring us closer to (but not yet at) the ultimate goal of optimal extractors for all settings of parameters [RRV99a, ISW00, RVW00, RSW00, TUZ01].

The work most directly related to ours is that of Hartman and Raz [HR00], which is not concerned with improving the parameters but rather the explicitness. Specifically, they show how to construct extractors with the same parameters as ours, but which are computable in *logarithmic space* rather than just polynomial time. They do this by giving a logarithmic-space construction of weak designs based on low-degree polynomials. (Our construction of weak designs, based on the Method of Conditional Expectations, appears inherently sequential.)

We remark that it might be possible to obtain extractors which extract all the randomness using a seed of length $O(\log^2 n)$ just by giving an improved construction of weak designs. This corresponds to the gap of $\Theta(\log 1/\gamma)$ between our upper and lower bounds for weak designs (Lemma 17 and Proposition 18). We leave closing this gap as an open problem.

Acknowledgments

While doing this work, the third author had many useful discussions with various people — including Oded Goldreich, Shafi Goldwasser, Adam Klivans, Madhu Sudan, Amnon Ta-Shma, Luca Trevisan, Avi Wigderson, and David Zuckerman. Some of these discussions led to specific improvements in this paper, which are not enumerated for the sake of brevity. Special gratitude goes to Luca Trevisan, for sharing his beautiful new insights into this area; and to Oded Goldreich for his never-ending supply of guidance and support, and all his help with the presentation. We also thank the anonymous referees for helpful comments on the presentation.

References

[AKSS89] Miklós Ajtai, János Komlós, William Steiger, and Endre Szemerédi. Almost sorting in one round. In Silvio Micali, editor, *Randomness and Computation*, volume 5 of *Advances in Computing Research*, pages 117–125. JAI Press Inc., 1989.

[AS00] Noga Alon and Joel H. Spencer. *The Probabilistic Method*. Interscience Series in Discrete Mathematics and Optimization. John Wiley and Sons, Inc., 2nd edition, 2000.

[ACR97] Alexander E. Andreev, Andrea E. F. Clementi, and José D. P. Rolim. Worst-case hardness suffices for derandomization: A new method for hardness-randomness trade-offs. In Pierpaolo Degano, Robert Gorrieri, and Alberto Marchetti-Spaccamela, editors, *Automata, Languages and Programming, 24th International Colloquium*, volume 1256 of *Lecture Notes in Computer Science*, pages 177–187, Bologna, Italy, 7–11 July 1997. Springer-Verlag.

[BGS98] Mihir Bellare, Oded Goldreich, and Madhu Sudan. Free bits, PCPs, and nonapproximability—towards tight results. *SIAM Journal on Computing*, 27(3):804–915, June 1998.

[CG88] Benny Chor and Oded Goldreich. Unbiased bits from sources of weak randomness and probabilistic communication complexity. *SIAM Journal on Computing*, 17(2):230–261, April 1988.

[CW89] Aviad Cohen and Avi Wigderson. Dispersers, deterministic amplification, and weak random sources (extended abstract). In *30th Annual Symposium on Foundations of Computer Science*, pages 14–19, Research Triangle Park, North Carolina, 30 October–1 November 1989. IEEE.

[CT91] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. Wiley Series in Telecommunications. John Wiley & Sons, Inc., 2nd edition, 1991.

[EFF85] Paul Erdős, Peter Frankl, and Zoltán Füredi. Families of finite sets in which no set is covered by the union of r others. *Israel Journal of Mathematics*, 51(1-2):79–89, 1985.

[GRS00] Oded Goldreich, Ronitt Rubinfeld, and Madhu Sudan. Learning polynomials with queries: The highly noisy case. *SIAM Journal on Discrete Mathematics*, 13, 2000.

[GW97] Oded Goldreich and Avi Wigderson. Tiny families of functions with random properties: A quality-size trade-off for hashing. *Random Structures & Algorithms*, 11(4):315–343, 1997.

[GZ97] Oded Goldreich and David Zuckerman. Another proof that BPP $\subseteq$ PH (and more). *Electronic Colloquium on Computational Complexity* Technical Report TR97-045, September 1997. `http://www.eccc.uni-trier.de/eccc`.

[HR00] Tzvika Hartman and Ran Raz. On the distribution of the number of roots of polynomials and explicit logspace extractors. In *Proceedings of RANDOM 2000*, number 8 in Proceedings in Informatics. Carleton Scientific, 14 July 2000.

[ILL89] Russell Impagliazzo, Leonid A. Levin, and Michael Luby. Pseudo-random generation from one-way functions (extended abstracts). In *Proceedings of the Twenty First Annual ACM Symposium on Theory of Computing*, pages 12–24, Seattle, Washington, 15–17 May 1989.

[ISW00] Russell Impagliazzo, Ronen Shaltiel, and Avi Wigderson. Extractors and pseudo-random generators with optimal seed length. In *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing*, pages 1–10, Portland,Oregan, 21–23 May 2000.

[IW96] Russell Impagliazzo and Avi Wigderson. An information theoretic variant of the inclusion-exclusion bound (preliminary version). Unpublished manuscript, September 1996.

[IW97] Russell Impagliazzo and Avi Wigderson. $P = BPP$ if E requires exponential circuits: Derandomizing the XOR lemma. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, pages 220–229, El Paso, Texas, 4–6 May 1997.

[MS77] Florence J. MacWilliams and Neil J. A. Sloane. *The Theory of Error-Correcting Codes*. North-Holland, 1977.

[MR95] Rajeev Motwani and Prabhakar Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995.

[Nis96] Noam Nisan. Extracting randomness: How and why: A survey. In *Proceedings, Eleventh Annual IEEE Conference on Computational Complexity*, pages 44–58, Philadelphia, Pennsylvania, 24–27 May 1996. IEEE Computer Society Press.

[NT98] Noam Nisan and Amnon Ta-Shma. Extracting randomness: A survey and new constructions. *Journal of Computer and System Sciences*, 1998. To appear in STOC '96 special issue. Preliminary versions in [Nis96] and [Ta-96].

[NW94] Noam Nisan and Avi Wigderson. Hardness vs randomness. *Journal of Computer and System Sciences*, 49(2):149–167, October 1994.

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

[Pip87] Nicholas Pippenger. Sorting and selecting in rounds. *SIAM Journal on Computing*, 16(6):1032–1038, December 1987.

[RT97] Jaikumar Radhakrishnan and Amnon Ta-Shma. Tight bounds for depth-two superconcentrators. In *38th Annual Symposium on Foundations of Computer Science*, pages 585–594, Miami Beach, Florida, 20–22 October 1997. IEEE.

[RR99] Ran Raz and Omer Reingold. On recycling the randomness of the states in space bounded computation. In *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing*, pages 159–168, Atlanta, Georgia, 1–4 May 1999.

[RRV99a] Ran Raz, Omer Reingold, and Salil Vadhan. Error reduction for extractors. In *40th Annual Symposium on Foundations of Computer Science*, pages 191–201, New York City, New York, 17–19 October 1999. IEEE.

[RRV99b] Ran Raz, Omer Reingold, and Salil Vadhan. Extracting all the randomness and reducing the error in Trevisan's extractors. In *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing*, pages 149–158, Atlanta, Georgia, 1–4 May 1999.

[RRV99c] Ran Raz, Omer Reingold, and Salil Vadhan. Extracting all the randomness and reducing the error in Trevisan's extractors. Technical Report TR99-046, Electronic Colloquium on Computational Complexity, 1999. `http://www.eccc.uni-trier.de/eccc`.

[RSW00] Omer Reingold, Ronen Shaltiel, and Avi Wigderson. Extracting randomness via repeated condensing. In *41st Annual Symposium on Foundations of Computer Science*, pages 22–31, Redondo Beach, CA, 17–19 October 2000. IEEE.

[RVW00] Omer Reingold, Salil Vadhan, and Avi Wigderson. Entropy waves, the zig-zag graph product, and new constant-degree expanders and extractors. In *41st Annual Symposium on Foundations of Computer Science*, pages 3–13, Redondo Beach, CA, 17–19 October 2000. IEEE.

[Röd85] Vojtěch Rödl. On a packing and covering problem. *European Journal of Combinatorics*, 6(1):69–78, 1985.

[RZ98] Alexander Russell and David Zuckerman. Perfect information leader election in $\log^* n + o(1)$ rounds. In *39th Annual Symposium on Foundations of Computer Science*, Palo Alto, California, 8–11 November 1998. IEEE.

[SSZ98] Michael Saks, Aravind Srinivasan, and Shiyu Zhou. Explicit OR-dispersers with polylogarithmic degree. *Journal of the ACM*, 45(1):123–154, January 1998.

[SV86] Miklos Santha and Umesh V. Vazirani. Generating quasi-random sequences from semi-random sources. *Journal of Computer and System Sciences*, 33(1):75–87, August 1986.

[Sip88] Michael Sipser. Expanders, randomness, or time versus space. *Journal of Computer and System Sciences*, 36(3):379–383, June 1988.

[SZ98] Aravind Srinivasan and David Zuckerman. Computing with very weak random sources. To appear in *SIAM Journal on Computing*, 1998. Preliminary version in *FOCS '94*.

[STV99] Madhu Sudan, Luca Trevisan, and Salil Vadhan. Pseudorandom generators without the XOR lemma. In *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing*, pages 537–546, Atlanta, Georgia, 1–4 May 1999. In joint session with *Complexity '99*.

[Ta-96] Amnon Ta-Shma. On extracting randomness from weak random sources (extended abstract). In *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing*, pages 276–285, Philadelphia, Pennsylvania, 22–24 May 1996.

[TS98] Amnon Ta-Shma. Personal communication, August 1998.

[Ta-98] Amnon Ta-Shma. Almost optimal dispersers. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing*, pages 196–202, Dallas, TX, May 1998. ACM.

[TUZ01] Amnon Ta-Shma, Christopher Umans, and David Zuckerman. Loss-less condensers, unbalanced expanders, and extractors. 2001. To appear in: Proceedings of the 33rd Annual ACM Symposium on Theory of Computing, 2001.

[Tre99] Luca Trevisan. Construction of extractors using pseudo-random generators. In *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing*, pages 141–148, Atlanta, Georgia, 1–4 May 1999.

[Uma99] Christopher Umans. Hardness of approximating σ_2^p minimization problems. In *40th Annual Symposium on Foundations of Computer Science*, pages 465–474, New York City, New York, 17–19 October 1999. IEEE.

[Vaz84] Umesh V. Vazirani. *Randomness, Adversaries, and Computation.* PhD thesis, University of California, Berkeley, 1984.

[Vaz87a] Umesh V. Vazirani. Efficiency considerations in using semi-random sources (extended abstract). In *Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing*, pages 160–168, New York City, 25–27 May 1987.

[Vaz87b] Umesh V. Vazirani. Strong communication complexity or generating quasirandom sequences from two communicating semirandom sources. *Combinatorica*, 7(4):375–392, 1987.

[VV85] Umesh V. Vazirani and Vijay V. Vazirani. Random polynomial time is equal to slightly-random polynomial time. In *26th Annual Symposium on Foundations of Computer Science*, pages 417–428, Portland, Oregon, 21–23 October 1985. IEEE.

[Weg87] Ingo Wegener. *The Complexity of Boolean Functions.* Wiley-Teubner Series in Computer Science. John Wiley & Sons, 1987.

[WZ99] Avi Wigderson and David Zuckerman. Expanders that beat the eigenvalue bound: explicit construction and applications. *Combinatorica*, 19(1):125–138, 1999.

[Yao82] Andrew C. Yao. Theory and applications of trapdoor functions (extended abstract). In *23rd Annual Symposium on Foundations of Computer Science*, pages 80–91, Chicago, Illinois, 3–5 November 1982. IEEE.

[Zuc96] David Zuckerman. Simulating BPP using a general weak random source. *Algorithmica*, 16(4/5):367–391, October/November 1996.

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

A Error-correcting codes

Proof of Lemma 7: The code we need can be obtained by "concatenating" a Reed-Solomon code with a Hadamard code (cf., [MS77]). Specifically, given parameters n and δ, let $\sigma = \delta^2$ and let $m = \lceil \log(n/\sigma) \rceil$. Let $\mathrm{Had} : \{0,1\}^m \to \{0,1\}^{2^m}$ be the *Hadamard code* — that is, for $x, y \in \{0,1\}^m$, the y'th component of $\mathrm{Had}(x)$ is the inner-product of x and y mod 2. Thus, for $x_1 \neq x_2$, $\mathrm{Had}(x_1)$ and $\mathrm{Had}(x_2)$ have (relative) Hamming distance 1/2. Let $F = \mathrm{GF}(2^m)$; an explicit description of F can be found in time $\mathrm{poly}(2^m) = \mathrm{poly}(n, 1/\delta)$ by exhaustively searching for an irreducible polynomial of degree m over GF(2). We can view strings $x \in \{0,1\}^n \subset (\{0,1\}^m)^{\lceil n/m \rceil}$ as giving the coefficients of a polynomial p_x of degree at most $d = \lceil n/m \rceil$ over F.

Now we define the error-correcting code $\mathrm{EC} : \{0,1\}^n \to (\{0,1\}^{2^m})^{|F|}$ as

$$\mathrm{EC}(x) = (\mathrm{Had}(p_x(a_1)), \ldots, \mathrm{Had}(p_x(a_{|F|}))),$$

where $a_1, \ldots, a_{|F|}$ are all the elements of F. Thus the codewords are of length $\overline{n} = 2^m \cdot |F| = O(n^2/\delta^4)$. Now we show that the (relative) minimum distance of this code is $1/2 - \sigma/2$. For any two distinct elements x and y of $\{0,1\}^n$, p_x and p_y disagree in at least $|F| - d$ elements F (as they are distinct polynomials of degree d). For each a such that $p_x(a) \neq p_y(a)$, $\mathrm{Had}(p_x(a))$ and $\mathrm{Had}(p_y(a))$ disagree in $2^m/2$ positions. Thus, for distinct x and y, $\mathrm{EC}(x)$ and $\mathrm{EC}(y)$ disagree in at least $q = (|F| - d) \cdot 2^m/2$ positions, for a relative distance of

$$\frac{q}{|F| \cdot 2^m} = \frac{1}{2} - \frac{d}{2|F|} \geq \frac{1}{2} - \frac{n}{2(n/\sigma)} = \frac{1}{2} - \frac{\sigma}{2}.$$

Now we apply the following general bound (cf., [BGS98, Lemma A.1]).

Lemma 35 *Suppose C is an error-correcting code with (relative) minimum distance $\geq 1/2 - \beta/2$. Then every Hamming ball of (relative) radius $1/2 - \sqrt{\beta}$ contains at most $1/3\beta$ codewords.*

Applying this lemma with $C = \mathrm{EC}$ and $\beta = \sigma$, we see that every Hamming ball of relative radius $1/2 - \delta$ has at most $1/3\delta^2 < 1/\delta^2$ codewords. ■