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

Ran Raz† Omer Reingold‡ Salil Vadhan§

November 16, 1999

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 from uniform distribution ε 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, Expander Graphs, Probabilistic Method, Pseudorandom Generators, Combinatorial Designs

*A preliminary version of this work appeared in *STOC '99* [RRV99b].

†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.

§MIT Laboratory for Computer Science. 545 Technology Square. Cambridge, MA 02139. USA. E-mail: salil@theory.lcs.mit.edu. URL: http://theory.lcs.mit.edu/~salil. During this work, the author was 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 improving the combinatorial construction underlying the Nisan–Wigderson generator used in Trevisan's construction. Applying a result of Wigderson and Zuckerman [WZ95] 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 $\text{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is called a (k,ε)-*extractor* if for every distribution X on $\{0,1\}^n$ of min-entropy k, the induced distribution $\text{Ext}(X, U_d)$ 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, 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 $\text{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ 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 $\{\text{Ext}_i : \{0,1\}^{n_i} \times \{0,1\}^{d_i} \to \{0,1\}^{m_i}\}_{i \in I}$ is called *explicit* if Ext_i can be evaluated in time $\text{poly}(n_i, d_i)$.

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 "Strong extractors and entropy loss."

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]. There have been a number of papers giving explicit constructions of dispersers and extractors, with a steady improvement in the parameters [Zuc96, NZ96, WZ95, GW97, SZ98, SSZ98, NT98, Zuc97, Ta-98, Tre99]. Most of the work on extractors is 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 that the Nisan–Wigderson pseudorandom generator [NW94] could be used to construct extractors.

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]; randomness-efficient error reduction in randomized algorithms and interactive proofs [Zuc97]; explicit constructions of expander graphs, superconcentrators, and sorting networks [WZ95]; hardness of approximation [Zuc96]; 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 are given in the following theorem:

Theorem 1 *For every $n, k, m \in \mathbb{N}$ and $\varepsilon > 0$, such that $m \leq k \leq n$, there are explicit (k, ε)-extractors* $\text{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *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 $m \leq k \leq n$, there are explicit (k, ε)-extractors* $\text{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *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 better discussed later in the context of strong extractors.

Plugging the second extractor of Theorem 1 into a construction of [WZ95] (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 $K \leq N$, 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 [WZ95, 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, WZ95], so our results translate similar improvements in each of these applications. We remark that the construction of [WZ95] used to obtain Corollary 3 requires extractors that extract nearly all the entropy of the source.

Techniques

Our work builds upon Trevisan's beautiful discovery of a 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 generator. This is the starting point for our work.

The Trevisan extractor. The Nisan–Wigderson generator [NW94] is a method of building a pseudorandom generator out of any Boolean function P such that the quality of the pseudorandom generator is closely related to how hard P is to compute (on average). Let $\mathcal{S} = (S_1, \ldots, S_m)$ be a collection of subsets of $[d]$, each 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 $\mathrm{NW}_{\mathcal{S},P} : \{0,1\}^d \to \{0,1\}^m$ is defined as

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

In the "indistinguishability proof" of [NW94], it is shown that for any function $D : \{0,1\}^m \to \{0,1\}$ which distinguishes the output of $\mathrm{NW}_{\mathcal{S},P}(y)$ (for uniformly selected y) from the uniform distribution on $\{0,1\}^m$, there is a small circuit C (or procedure of small "description size") such that $C^D(\cdot)$ (*i.e.* C with oracle access to D) approximates $P(\cdot)$ reasonably well. It is shown that the size of the C is related to $\max_{i \neq j} |S_i \cap S_j|$, so one should use a collection of sets in which this quantity is small, while trying to minimize the seed length d.

We now give a rough description of the Trevisan extractor $\textsc{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$. For a string $u \in \{0,1\}^n$, let $\overline{u} \in \{0,1\}^{\overline{n}}$ be an encoding of u in an error-correcting code and define $\ell = \log \overline{n}$. We view $\overline{u}$ as a Boolean function $\overline{u} : \{0,1\}^\ell \to \{0,1\}$. As above, we fix a collection $\mathcal{S} = (S_1, \ldots, S_m)$ of subsets of $[d]$ of size ℓ.

Then the extractor is simply

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

The analysis of this extractor in [Tre99] shows that the output of this extractor is close to uniform as long as the source min-entropy is greater than the size of the circuit C built in the indistinguishability proof of [NW94]. Hence, one needs to make sure this circuit size is not much larger than the number m of output bits while minimizing the number d of truly random bits needed, which is equal to the seed length of the Nisan–Wigderson generator.

Our improvements. The first improvement of this paper stems from the observation that actually $\max_i \sum_{j<i} 2^{|S_i \cap S_j|}$ is much better than $\max_{i \neq j} |S_i \cap S_j|$ as a measure of the size of the circuit built in the Nisan–Wigderson indistinguishability proof. So we are left with the problem of constructing set systems in which this quantity is small; we call such set systems *weak designs* (in contrast to *designs*, in which $\max_{i \neq j} |S_i \cap S_j|$ is bounded). We show that with weak designs, one can have d much smaller than is possible

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

with the corresponding designs. 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., [ASE92] 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.

The second improvement 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. In Trevisan's analysis for the size of the circuit C, the fact that $\overline{u}$ is an error-correcting code (rather than just an arbitrary function) is not used. The circuit complexity of the function $\overline{u}$, restricted to the subset of inputs $S_i \cap S_j$, is hence bounded by $\approx O(2^{|S_i \cap S_j|})$. Sometimes, however, this is a very bad upper bound. For example, the circuit complexity of the function $\overline{u}$ itself (without restriction) is $\approx O(2^n)$ which is sometimes much smaller than $O(2^{\overline{n}})$. This gap is significant when ε is relatively small (because small ε requires an error-correcting code with very good distance properties, which in turn requires long codewords.) Here, we suggest that if one uses multilinear error-correcting codes and constructs the weak designs appropriately then the circuit complexity of the function $\overline{u}$, restricted to the subset of inputs $S_i \cap S_j$, can be bounded by a value much smaller than $2^{|S_i \cap S_j|}$.

Strong extractors and entropy loss

Since a (k, ε)-extractor $\text{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ 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 $\text{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is called a *strong* (k, ε)*-extractor* if for every distribution X on $\{0,1\}^n$ of min-entropy k, the induced distribution $(U_d, \text{Ext}(X, U_d))$ 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 $k \leq n$, there exist strong extractors $\text{Ext}_{n,k} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{k-\Delta}$ 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 what is known, every entry in Figure 2 yields a (not necessarily strong) 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 $\text{polylog}\, n$. 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., $k = \text{polylog}\, n$), as d is linear in k. (The fact that d does not explicitly depend on ε here is not a contradiction, as no nontrivial extraction is occuring 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 $k \leq n$, there are explicit strong (k, ε)-extractors $\text{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{k-\Delta}$ 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. $d = O\left(\log^2 n \cdot \log(1/\varepsilon) \cdot \log k\right)$.

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
[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]	$d = O(\log(n/\varepsilon))$	$\Delta = \text{polylog}(n/\varepsilon)$	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

Actually, the method we use to reduce the entropy loss of our extractor is quite general, and can also be used to reduce the entropy losses of the extractors of [GW97] and [NT98] to $2\log(1/\varepsilon) + O(1)$ while only increasing the number of truly random bits used by a constant factor.

Organization

In Section 2, we introduce the notion of a weak design, and state our results on constructing them. We also state 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 3, where we analyze Trevisan's extractor when our weak designs are used. In Section 4, we sketch our method for improving the dependence on the error as claimed in Theorem 2. Section 5 contains our construction of weak designs via the Probabilistic Method, and our lower bounds for designs are proven in Section 6. In Section 7, we argue that our extractors are actually strong extractors, and show how to achieve optimal entropy loss. In Section 8, 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.

2 Combinatorial designs

The combinatorial construction underlying the Nisan–Wigderson generator are combinatorial designs.

Definition 5 ([NW94]) [4] *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 $i \neq j$, $|S_i \cap S_j| \leq \log \rho$.*

[4] There is a somewhat related notion in the combinatorics literature known as a *2-design* (see, e.g. [AK92]). In 2-designs, strong additional regularity requirements are imposed (such as all the pairwise intersections being *exactly* the same size and all points being contained in the same number of sets). These additional requirements are irrelevant in our applications.

Motivation. In Trevisan's extractor, 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.

One explicit construction of designs is given by the following:

Lemma 6 ([NW94, Tre99]) *For every $m, \ell \in \mathbb{N}$, and $\rho > 1$, there exists an efficiently constructible (ℓ, ρ)-design $S_1, \ldots, S_m \subset [d]$ with*

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

Notice that the dependence 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, proved in Section 6:

Proposition 7 *If $S_1, \ldots, S_m \subset [d]$ is an (ℓ, ρ)-design, then*

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

A slightly weaker bound (without the $(\ell - \log \rho)$ factor) can be deduced from the Ray-Chaudhuri–Wilson Theorem [BF92].

The first improvement of this paper stems from the observation that actually a weaker form of design suffices for the Nisan–Wigderson generator and the construction of extractors:

Definition 8 *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,*
$$\sum_{j<i} 2^{|S_i \cap S_j|} \leq \rho \cdot (m-1).$$

We will show that the parameters of a weak design correspond to the parameters of our extractors in the same way that designs corresponded to the parameters of Trevisan's extractor. Notice that every (ℓ, ρ)-design is a weak (ℓ, ρ)-design. But one can, for many settings of m, ℓ, and ρ, achieve weak (ℓ, ρ)-designs $S_1, \ldots, S_m \subset [d]$ with much smaller values of d than possible with (ℓ, ρ)-designs. Indeed, we will prove the following in Section 5 using a probabilistic argument:

Lemma 9 *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 $\mathrm{poly}(m, d)$.

This is already much better than what is given by Lemma 6; for constant ρ, 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 9 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 $\sum_{j<i} 2^{|S_i \cap S_j|} < \rho \cdot (i-1)$.[5]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 9 (directly inspired by the iteration of Wigderson and Zuckerman [WZ95] on extractors), and yields the following improvement.

Lemma 10 *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 is 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 $\mathrm{poly}(m, d)$.

The "in particular" part of Lemma 10 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 10 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. Lemma 10 will be proven in Section 5.

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, proved in Section 6, shows that our weak designs are optimal up to the $\log(1/\gamma)$ factor in our second construction.

Proposition 11 *For every (ℓ, ρ)-weak design $S_1, \ldots, S_m \subset [d]$,*

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

Notice that $d = m\ell$ can be trivially 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 $\Omega(\ell^2)$.

3 The extractor

In this section, we describe the Trevisan extractor and analyze its performance when used with our weak designs. The description of the extractor follows [Tre99] very closely. The main tool in the Trevisan extractor is the Nisan–Wigderson 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 12 (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.*

[5]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 21.

We can now describe the Trevisan extractor, which takes as parameters $n, m, k \in \mathbb{N}$, and $\varepsilon > 0$, where $m \leq k \leq n$. Let $\mathrm{EC} : \{0,1\}^n \to \{0,1\}^{\overline{n}}$ be as in Lemma 12, 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 $\textsc{Ext}_{\mathcal{S}} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is defined as

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

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

Lemma 13 ([Yao82]) *Suppose that Z is a distribution on $\{0,1\}^m$ whose statistical difference from U_m is greater than ε. Then there is an $i \in [m]$ and a function ("next-bit predicator") $A : \{0,1\}^{i-1} \to \{0,1\}$ such that*

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

Moreover, if there is a circuit $D : \{0,1\}^m \to \{0,1\}$ of size s distinguishing Z from 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 13 in the analysis of our extractor; it will only be used for our quantitative improvement to the pseudorandom generators of [NW94] given in Section 8.

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

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

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

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

2. $\log |\mathcal{F}_i| \leq \sum_{j<i} 2^{|S_i \cap S_j|}$.

3. *Each function in $\mathcal{F}_i$ can be computed by a circuit of size $O\left(\sum_{j<i} |S_i \cap S_j| \cdot 2^{|S_i \cap S_j|}\right)$.*[6]

The improvement over [NW94, Tre99] in Lemma 14 is the use of $\sum_{j<i} 2^{|S_i \cap S_j|}$ rather than $m \cdot 2^{\max_i |S_i \cap S_j|}$ 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 8.

Proof: Let

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

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 $j \neq i$ is now a function P_j of x that depends on only $|S_i \cap S_j|$ bits of x. So, we have

$$\Pr_x\left[A\left(P_1(x) \cdots P_{i-1}(x)\right) = P(x)\right] \geq \alpha.$$

[6] 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 $x \mapsto (P_1(x), P_2(x), \ldots, P_{i-1}(x))$, 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 $2^{|T_{ij}|} = 2^{|S_i \cap S_j|}$. So, the total number of bits it takes to represent a function in $\mathcal{F}_i$ is at most $\sum_{j<i} 2^{|S_i \cap S_j|}$, 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 k bits can be computed by a circuit of size $O(k2^k)$. ■

We now analyze the extractor $\text{EXT}_{\mathcal{S}}$ 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 14.

Proposition 15 *If $\mathcal{S} = (S_1, \ldots, S_m)$ (with $S_i \subset [d]$) is a weak (ℓ, ρ)-design for $\rho = (k - 3\log(m/\varepsilon) - 3)/m$, then $\text{EXT}_{\mathcal{S}} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is a (k, ε)-extractor.*

The proof of Proposition 15 basically follows the analysis of Trevisan's extractor in [Tre99] except that we use the more refined bounds on $|\mathcal{F}_i|$ given by Lemma 14.

Proof: Let X be any distribution of min-entropy k. We need to show that the statistical difference between U_m and $\text{EXT}(X, U_d)$ is at most ε. By Lemma 13, it suffices to show that for every next-bit predictor $A\colon \{0,1\}^{i-1} \to \{0,1\}$,

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

where y is selected uniformly from $\{0,1\}^d$. So let $A\colon \{0,1\}^{i-1} \to \{0,1\}$ be any next-bit predictor and let $\mathcal{F}_i$ be as in Lemma 14, so that $|\mathcal{F}_i| \leq 2^{\rho m}$.

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 12, 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,

$$|B| \leq (2m/\varepsilon)^2 \cdot |\mathcal{F}_i| \leq (2m/\varepsilon)^2 \cdot 2^{\rho m}.$$

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

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

Now, by Lemma 14, if $u \notin B$, then

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

Thus,

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

■

Combining Proposition 15 with the weak designs given by Lemmas 9 and 10 essentially proves Theorem 1. The only technicality is that Proposition 15 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 $\rho = (k-\Delta)/m$ and consequently lose $\Delta = 3\log(m/\varepsilon) + 3$ bits of the source entropy in Proposition 15. However, since Δ is so small, we can give our extractor Δ more truly random bits in its seed (increasing d by only a $o(d)$ additive term) which we just concatenate to the output to compensate for the loss. The details of this are given below.

Proof of Theorem 1: Let $\Delta = 3\log(m/\varepsilon)+3$. Let $k' = k-\Delta$, $m' = m-\Delta-1$, and $\rho = k'/m' > k/(m-1)$. For Part 1 (resp., Part 2), apply Proposition 15 with the weak (ℓ,ρ)-design $S_1,\ldots,S_{m'} \subset [d']$ of Lemma 9 (resp., Lemma 10). This gives an (k,ε)-extractor $\text{EXT}: \{0,1\}^n \times \{0,1\}^{d'} \to \{0,1\}^{m'}$, with $d' = O\left(\frac{\log^2(n/\varepsilon)}{\log(k/m)}\right)$ (resp., $d' = O(\log^2(n/\varepsilon)\log(1/\gamma))$). By using $\Delta+1$ additional bits in the seed and simply concatenating these to the output, we obtain a (k,ε)-extractor $\text{EXT}: \{0,1\}^n \times \{0,1\}^{d'+\Delta+3} \to \{0,1\}^m$, as desired. (In applying Lemma 10, we need to make sure that $\rho < 3/2$, but if $\rho \geq 3/2$, we can use the weak design of Lemma 9 instead.) ■

Remark 16 A finer analysis can improve Proposition 15 to use $\rho = (k - 2\log(1/\varepsilon) - 3\log m - 4)/m$ rather than $\rho = (k - 3\log(m/\varepsilon) - 3)/m$. 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.

4 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 11). 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 $\text{poly}(n,\varepsilon)$, 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 14) will be built from functions of the form $P_j : F^{|S_i \cap S_j|} \to F$. In the proof of Lemma 14, 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 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 3.

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 unpredictable given the first $i-1$ components means that the output is what is known as a *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 $\textsc{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ as a function

$$\textsc{Ext} : F^{n'} \times F^{d'} \to F^{m'},$$

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. 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 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 $\textsc{Ext}_{\mathcal{S}} : F^{n'} \times F^{d'} \to F^{m'}$ as

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

(The function $\textsc{Ext}$ is still not our final extractor). Note that the number of truly random bits used by $\textsc{Ext}$ is $d' \log|F| = O(d' \cdot \log(1/\varepsilon))$. The following lemma is analogous to Lemma 14. 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 18 *Fix $\mathcal{S}$. For every $i \in [m']$, there is a set $\mathcal{F}_i$ of functions from F^ℓ to F^{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 $A : F^{i-1} \to F$, there exists a function $f \in \mathcal{F}_i$ such that*

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

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

2. $\log|\mathcal{F}_i| \leq \sum_{j<i} 2^{|S_i \cap S_j|} \cdot \log|F|$.

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 14.

Now assume that $\mathcal{S}$ is a weak (ℓ, ρ)-design for $\rho = (k - c \cdot \log |F|)/m$ (where, say, $c = 10$), and let X be any distribution of min-entropy k. The following proposition shows that $\text{EXT}(X, U_d)$ doesn't have a good next-element predictor. The proposition is analogous to Proposition 15.

Proposition 19 *If $\mathcal{S} = (S_1, \ldots, S_{m'})$ (with $S_i \subset [d']$) is a weak (ℓ, ρ)-design for $\rho = (k - c \cdot \log |F|)/m$ (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 $A : F^{i-1} \to F$,*

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

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 15, except that we use the distance property of multilinear error-correcting codes given by Lemma 17 and we use Lemma 18 rather than Lemma 14.

In general, the function $\text{EXT}_{\mathcal{S}}$ is not a good extractor. Nevertheless, by Proposition 19, we know that each element $\text{EXT}(X, U_d)$ has large min-entropy ($\delta \log |F|$ bits) given all its predecessors. That is, it is a "block-wise source" in the sense of [CG88], in which the min-entropy of each block given the predecessors is a constant fraction of its length (which is $\log |F|$). We can now construct an extractor from $\text{EXT}_{\mathcal{S}}$ in one of the following ways:

1. By applying on the entire output $\text{EXT}(X, U_d)$ 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 $\text{EXT}(X, U_d)$ 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).

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 Lemmas 9 (as in the proof of Theorem 1). The resulting seed length (using an (ℓ, ρ)-weak design for $\rho = (k - c \log |F|)/m$) 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 upto a small constant fraction of the min-entropy (even if we use the weak designs of Lemma 10). In order to extract more of the min-entropy of the source, we will need to use iterations, as in [WZ95] (cf., Lemma 27). 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.

5 Construction of weak designs

Proof of Lemma 9: 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. $\sum_{j<i} 2^{|S_i \cap S_j|} \leq \rho \cdot (i - 1)$.

Supppose 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 [ASE92] (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 [ASE92] and [MR95, Ch. 5]). Details can be found in Appendix B. ■

Remark 20 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.

Proof of Lemma 10: Let $d_0 = \lceil \ell/\ln 2 \rceil \cdot \ell$, $h = \lceil \log(2/\gamma) \rceil$, 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 $n_t = \lfloor (1 - 2^{-t}) \cdot (1+\gamma) \cdot m \rfloor$ 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 9. 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 $\geq m$, 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

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

as desired. ■

6 Lower bounds for designs

Proof of Proposition 7: Let $I = \max_{i \neq j} |S_i \cap S_j| \leq \log \rho$. For each $j = 1, \ldots, m$, let Γ_j be the set of subsets of S_j of size $I+1$, so $|\Gamma_j| = \binom{\ell}{I+1}$. Let $\Gamma = \bigcup_j \Gamma_j$. Notice that the sets Γ_j are disjoint, because no two distinct sets S_i, S_j share more than I elements. Thus, $|\Gamma| = m \cdot \binom{\ell}{I+1}$. At the same time, $|\Gamma|$ consists of subsets of $[d]$ of size $I+1$, so $|\Gamma| \leq \binom{d}{I+1}$. So we have

$$m \cdot \binom{\ell}{I+1} \leq \binom{d}{I+1}.$$

Expanding the binomial coefficients and rearranging terms, we have

$$m \leq \left(\frac{d}{\ell}\right)\left(\frac{d-1}{\ell-1}\right)\cdots\left(\frac{d-I}{\ell-I}\right) \leq \left(\frac{d}{\ell-I}\right)^{I+1} \leq \left(\frac{d}{\ell-\log\rho}\right)^{\log 2\rho}$$

■

Proof of Proposition 11: Let $\overline{m} = 2d/\ell$. If $\overline{m} \geq m$, we are done, so we may assume that $\overline{m} < m$. We will now consider only the first $\overline{m}$ sets.[7] We have

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

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

$$\log 2\rho > \frac{2}{\overline{m}^2} \sum_{j<i\leq\overline{m}} |S_i \cap S_j| \tag{1}$$

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 1, we have

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

which proves the proposition. ■

[7] In this proof, we are assuming that $\overline{m} = 2d/\ell$ is an integer, but, with a slightly more tedious proof, it is possible to obtain exactly the same bound without this assumption.

Remark 21 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 9). If we have such a bound, then summing over i from 1 to $\overline{m}$ gives

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

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 1. Following the rest of the proof without change, this shows that

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

Remark 22 Using an information-theoretic analogue of the inclusion-exclusion bound, due to Impagliazzo and Wigderson [IW96], one can generalize the lower bound of Proposition 11 to a wider class of generators with similar properties to the Nisan–Wigderson generator. Specifically, one can prove the following:

Proposition 23 *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

$$\mathrm{H}(X) \geq \min\left(\frac{4\ell^2}{9\log 2\rho}, \frac{m\ell}{2}\right).$$

In Proposition 23, $\mathrm{H}(\cdot)$ 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 23 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 11, replacing the usual inclusion-exclusion bound with that of [IW96], which states that $\mathrm{H}(X) \geq \sum_i \mathrm{H}(X_i|Y_i) - \sum_{j<i} \mathrm{H}(X_j|Y_i)$.[8]

To compare Proposition 23 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 23 are even weaker.

7 Strong extractors and entropy loss

Recall that a *strong (k,ε)-extractor* is a function $\textsc{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ such that 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 $U_d \times U_m$. Also recall that the *entropy loss* of a strong extractor is defined to be $\Delta = k - m$, and that nonconstructively it is possible to have $\Delta = 2\log(1/\varepsilon) + O(1)$ with $d = \log(n-k) + 2\log(1/\varepsilon) + O(1)$.

[8] The slightly worse constant 4/9 appears instead of 1/2 since we must make sure that $\overline{m}$ is an integer (eg., take $\overline{m} = \lceil 2d/\ell \rceil$) and in this case, we don't know an alternative proof that avoids this problem.

In this section, we observe that our extractors are in fact strong extractors. Then we show how to achieve optimal entropy loss while preserving the the number of truly random bits used up to a constant factor. To show that our extractors $\textsc{Ext}_{\mathcal{S}}(u, y) = \mathrm{NW}_{\mathcal{S},\overline{u}}(y)$ are actually strong extractors, we simply note that the analysis of the Nisan–Wigderson generator goes through essentially unchanged even if the seed y is revealed to the distinguisher. We obtain the following strong-extractor analogue of Proposition 15:

Proposition 24 *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 $\textsc{Ext}_{\mathcal{S}} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is a strong (k, ε)-extractor.*

Proof sketch: Let X be any distribution on $\{0,1\}^n$ of min-entropy k. Suppose that $Z = (U_d, \textsc{Ext}_{\mathcal{S}}(X, U_d))$ has statistical difference greater than ε from $U_d \times U_m$. An analogue of Lemma 13 which uses the fact that the first d bits of Z are distributed uniformly says that there is an $i \in [m]$ and a function $A : \{0,1\}^d \times \{0,1\}^{i-1} \to \{0,1\}$,

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

where y is selected uniformly from $\{0,1\}^d$. Now the analysis proceeds just as the proof of Proposition 15, except that we need an analogue of Lemma 14 for predictors A that are also given y as input. To obtain such an analogue, we must simply count functions of the form $y|_{S_i} \mapsto (y, P(y|_{S_1}), \ldots, P(y|_{S_{i-1}}))$ rather than those of the form $y|_{S_i} \mapsto (P(y|_{S_1}), \ldots, P(y|_{S_{i-1}}))$ (where all the bits of $y \in \{0,1\}^d$ outside S_i are fixed). To describe the first component y of such a mapping requires an additional $d - \ell \le d$ bits (to specify the fixed bits of y outside of S_i). Thus, we use the bound $\log |\mathcal{F}_i| \le d + \sum_{j<i} 2^{|S_i \cap S_j|}$ in place of Item 2 in Lemma 14. □

Remark 25 The extra d bits of entropy loss caused by the weaker bound on $|\mathcal{F}_i|$ can be saved if the analysis is done slightly differently. Specifically, the bits of y outside S_i can be fixed at the same time as when Lemma 13 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. We omit additional details of how to save these d bits lost by the weaker analysis, as they will be easily regained by the method given below for achieving optimal entropy loss.

Combining Proposition 24 and Lemma 10 with $m = k - O(\log^2(n/\varepsilon) \cdot \log k)$ gives the following:

Proposition 26 *For every n, k, and ε such that $k \le n$, there is an explicit strong (k, ε)-extractor $\textsc{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{k-\Delta}$ with*

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

and entropy loss $\Delta = O(d)$.

By following Remarks 16 and 25, the entropy loss can be reduced to $2\log(1/\varepsilon) + 3\log k + O(1)$, but we will use a different method to make the entropy loss even better. We use an idea due to Wigderson and Zuckerman [WZ95]: 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 [WZ95]:

Lemma 27 *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 [WZ95] is that the statistical difference from uniform in Ext has a better dependence on s (in [WZ95], the expression is $\varepsilon_1 + \varepsilon_2 + 2^{-s}$). There is also an analogue of Lemma 27 for non-strong extractors; in that case, 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 27, let us see how we can use it to make our entropy loss optimal. If we use the extractor given by Proposition 26 as Ext_1, Lemma 27 tells us that we need only find an extractor Ext_2 which works well for very small (i.e., polylogarithmic) min-entropy. The following "low min-entropy" extractor of Srinivasan and Zuckerman [SZ98] achieves exactly what we want:

Lemma 28 ([SZ98]) *For every n, $k \leq n$, and $\varepsilon > 0$, there is an explicit strong (k, ε)-extractor $\text{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{k-\Delta}$ with entropy loss $\Delta = 2\lceil \log(1/\varepsilon) \rceil + 2$ and $d = O(k + \log n)$.*

Now we prove Theorem 4:

Proof of Theorem 4: Let n, $k \leq n$, and $\varepsilon > 0$ be given. By Proposition 26, there is an explicit $(k, \varepsilon/4)$-extractor $\text{Ext}_1 : \{0,1\}^n \times \{0,1\}^{d_1} \to \{0,1\}^{m_1}$ with $d_1 = O((\log^2(n/\varepsilon))(\log k))$ and entropy loss $\Delta_1 = O(d_1)$. By Lemma 28, there is an explicit $(\Delta_1 - 1, \varepsilon/2)$-extractor $\text{Ext}_2 : \{0,1\}^{n+d_1} \times \{0,1\}^{d_2} \to \{0,1\}^{m_2}$ with $d_2 = O(\Delta_1 + \log n) = O(d_1)$ and entropy loss $\Delta_2 = 2\lceil \log(2/\varepsilon) \rceil + 2 = 2\lceil \log(1/\varepsilon) \rceil + 4$. Combining these two extractors via Lemma 27 (with $s = 1$), there is an explicit (k, ε)-extractor $\text{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ with $d = d_1 + d_2 = O(\log^2(n/\varepsilon) \cdot \log k)$ and entropy loss $\Delta = \Delta_2 + 1 = 2\lceil \log(1/\varepsilon) \rceil + 5$. Part 2 of Theorem 4 is proven by applying the same observations to the extractors of Theorem 2. That is, we observe, analogous to Proposition 24, that they are actually strong extractors and then use Lemmas 27 and 28 to reduce the entropy loss to optimal. ■

We note that the same method can be used to reduce the entropy losses of the extractors of [GW97] and [NT98] to $2\log(1/\varepsilon) + O(1)$, while preserving the number of truly random bits used up to a constant factor. (For [NT98], one should use the version of Lemma 27 for non-strong extractors and the resulting extractor will not be strong.)

Remark 29 It is possible to achieve entropy loss $2\log(1/\varepsilon) + O(1)$ with just the extractors in this paper, without using the extractor of [SZ98]. Following Remark 16, we obtain a version of Proposition 24 in which $\rho = (k - 2\log(1/\varepsilon) - 3\log m - O(1))/m$. In particular, we have for any $k > 2\log(1/\varepsilon) + O(1)$, a strong (k, ε)-extractor $\text{Ext}^* : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^2$ with $d = O(\log^2(n/\varepsilon))$. We can repeatedly use this extractor (with careful settings of k and ε) as Ext_2 in Lemma 27 to reduce the entropy loss of the extractor given by Proposition 26 (i.e., we start with the extractor of Proposition 26 as Ext_1 and obtain an extractor with smaller entropy loss which becomes Ext_1 in the next application of Proposition 26). The fact that Ext^* works for any $k > 2\log(1/\varepsilon) + O(1)$ allows us to keep reducing the entropy loss until it drops to $2\log(1/\varepsilon) + O(1)$.

Proof of Lemma 27: 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 $\Pr[\text{Ext}_1(X, y) = z] < 2^{-(m_1+s)}$ (i.e., L is the set of pairs (y, z) for which z is "light" under $\text{Ext}_1(\cdot, y)$, 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 30 *For every $(y, z) \notin L$, the conditional distribution of X given that $\text{Ext}_1(X, y) = z$ has min-entropy at least $\Delta_1 - s$.*

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

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

This proves Claim 30. □

Thus, since EXT_2 is a $(\Delta_1 - s, \varepsilon_2)$-extractor, for every $(y,z) \notin L$, the conditional distribution of $(U_{d_2}, \text{EXT}_2(X, U_{d_2}))$ given that $(U_{d_1}, \text{EXT}_1(X, U_{d_1})) = (y,z)$ has statistical difference at most ε_2 from uniform. Now we argue that $(U_{d_1}, \text{EXT}_1(X, U_{d_1}))$ lands in L with low probability.

Claim 31 $\Pr[(U_{d_1}, \text{EXT}_1(X, U_{d_1})) \in L] \leq \varepsilon_1/(2^s - 1)$.

> **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, $\Pr[U_{d_1} \times U_{m_1} \in L] > 2^s \cdot \Pr[(U_{d_1}, \text{EXT}_1(X, U_{d_1})) \in L]$. 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
>
> $$\Pr[U_{m_1} \in L] - \Pr[\text{EXT}_1(X, U_{d_1}) \in L] \geq (2^s - 1) \cdot \Pr[\text{EXT}_1(X, U_{d_1}) \in L].$$
>
> Since EXT_1 is a strong (k, ε_1)-extractor, this statistical difference is at most ε_1, and the claim follows. □

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, 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, 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 27. ■

8 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 14, we see that the relevant notion of design in the setting of pseudorandom generation versus small circuits is the following:

Definition 32 *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,*
$$\sum_{j<i} |S_i \cap S_j| \cdot 2^{|S_i \cap S_j|} \leq \rho \cdot (m-1).$$

Notice that it is meaningful to consider even values of ρ less than 1, since $|S_i \cap S_j| \cdot 2^{|S_i \cap S_j|}$ can be zero. Using a construction like the one in Lemma 9, we obtain

Lemma 33 *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*

$$d = \begin{cases} O\left(\frac{\ell^2}{\ln \rho \ell}\right) & \text{if } \rho \geq \frac{6}{\ell} \\ O\left(\frac{\ell}{\rho}\right) & \text{if } \rho < \frac{6}{\ell} \end{cases}$$

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

The quantitative relation between pseudorandom generators and type 2 weak designs follows readily from Lemmas 13 and 14:

Lemma 34 *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 $s - O(\rho m)$ can distinguish $\mathrm{NW}_{\mathcal{S},P}$ from uniform with advantage greater than $m\varepsilon$.*

Combining this and Lemma 33 with $s = 2m$ and ρ a small constant, we obtain

Theorem 35 *Suppose $P\colon \{0,1\}^\ell \to \{0,1\}$ is a predicate such that 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 $G_{P,m}\colon \{0,1\}^{O(\ell^2/\log \ell)} \to \{0,1\}^m$ 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 $O(\ell^2)$).

9 Conclusions

We have shown how to improve Trevisan's extractor construction when using it to extract most or all of the randomness from a weak random source or when requiring the error of the extractor to be very small. These results bring us closer to the ultimate goal of having "optimal" explicit extractors for all settings of parameters, while maintaining the directness and simplicity of Trevisan's original construction.

One place where our construction could potentially be improved is the construction of weak (ℓ,ρ)-designs when $\rho = 1$, which corresponds to the case when one wants to extract *all* the randomness from the weak random source. There is a gap of $\log m$ between the upper and lower bounds in this case (Lemma 10 and Proposition 11, respectively). However, this is the only improvement that is possible by just modifying the construction of weak designs; that is, $\Omega(\log^2 n)$ is a barrier for extracting even a constant fraction of the randomness using just the current analysis with weak designs.

Still, there may be room for other kinds of improvements to the Nisan–Wigderson generator and Trevisan's construction. One such improvement has been recently given by [ISW99b, ISW99a], who show how to build extractors with seed length $O(\log n)$ for *any* min-entropy k (albeit while extracting a sublinear amount of the randomness); recall that Trevisan's construction uses $O((\log^2 n)/(\log k))$ truly random bits to extract $k^{1-\alpha}$ bits, which is only logarithmic when $k = n^{\Omega(1)}$.

The definition of extractors assumes that one knows in advance how much randomness is in the weak random source. It is natural to ask what one can achieve if this is not the case. One reasonable requirement is that there should be one extractor for all min-entropies, and the extractor should have the following property: if the source has min-entropy k, then the first k (or, more generally, $m(k)$) bits of the output wil be distributed almost uniformly. It turns out that our construction (as well as Trevisan's original construction) can yield extractors with this property. The reason is that our construction of a weak $(\ell, 2)$-design $(S_1, ..., S_m)$ has the property that, for every i, $(S_1, ..., S_i)$ is a also a weak $(\ell, 2)$-design. It would be interesting to see if this property can be exploited in any applications.

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 neverending supply of guidance and support, and all his help with 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.

[ASE92] Noga Alon, Joel H. Spencer, and Paul Erdős. *The Probabilistic Method*. Wiley-Interscience Series in Discrete Mathematics and Optimization. John Wiley and Sons, Inc., 1992.

[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.

[AK92] E.F. Assmus and J.D. Key. *Designs and their codes*. Number 103 in Cambridge Tracts in Mathematics. Cambridge University Press, 1992.

[BF92] László Babai and Péter Frankl. *Linear Algebra Methods in Combinatorics, with applications to Geometry and Computer Science*. University of Chicago, Department of Computer Science, September 1992.

[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.

[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`.

[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.

[ISW99a] Russell Impagliazzo, Ronen Shaltiel, and Avi Wigderson. Extractors and pseudo-random generators with optimal seed length. Manuscript, October 1999.

[ISW99b] Russell Impagliazzo, Ronen Shaltiel, and Avi Wigderson. Near-optimal conversion of hardness into pseudo-randomness. In *40th Annual Symposium on Foundations of Computer Science*, pages 181–190, New York City, New York, 17–19 October 1999. IEEE.

[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.

[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.

[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.

[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.

[WZ95] Avi Wigderson and David Zuckerman. Expanders that beat the eigenvalue bound: Explicit construction and applications. Technical Report CS-TR-95-21, University of Texas Department of Computer Sciences, 1995. To appear in *Combinatorica*.

[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 12: The code we need can be obtained by "concatenating" a Reed-Soloman 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 $\mathrm{GF}(2)$. 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 36 *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. ■

B Derandomizing the proof of Lemma 9

In the analysis of the probabilistic choice of S_i, we showed that

$$\mathrm{E}\left[\sum_{j<i} 2^{|S_i \cap S_j|}\right] \leq \rho \cdot (i-1)$$

By averaging, this implies that there exists an $\alpha_1 \in B_1$ such that

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

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 2 hold. Then, fixing such an α_1, another averaging argument implies that there exists an $\alpha_2 \in B_2$ such that

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

Again, assuming that we can compute the appropriate conditional expectations, we can find an α_2 that makes Inequality 3 hold. Proceeding like this, we obtain $\alpha_1, \ldots, \alpha_\ell$ such that

$$\mathrm{E}\left[\sum_{j<i} 2^{|S_i \cap S_j|} \middle| a_1 = \alpha_1, a_2 = \alpha_2, \ldots, a_\ell = \alpha_\ell\right] \leq \rho \cdot (i-1) \tag{4}$$

But now there is no more randomness left in the experiment, and Inequality 4 simply says that $\sum_{j<i} 2^{|S_i \cap S_j|} \leq \rho \cdot (i-1)$, 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 9 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.