Electronic Colloquium on Computational Complexity, Revision 1 of Report No. 47 (1998)

Extracting All the Randomness from a Weak Random Source*

Salil P. Vadhan†
Laboratory for Computer Science, MIT
Cambridge, MA 02139
http://www-math.mit.edu/~salil
salil@math.mit.edu

December 10, 1998

Abstract

We give explicit constructions of extractors which work for a source of any min-entropy on strings of length n. The first construction extracts any constant fraction of the min-entropy using $O(\log^2 n)$ additional random bits. The second extracts 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.

These 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.

Most of our results have been independently obtained by Ran Raz and Omer Reingold.

Keywords: Extractors, Expanders, Probabilistic Method, Pseudorandom Generators

*An earlier version of this paper appeared as MIT Laboratory for Computer Science Technical Memo MIT-LCS-TM-585.

†Supported by a DOD/NDSEG fellowship and partially by DARPA grant DABT63-96-C-0018.

ISSN 1433-8092

1 Introduction

Roughly speaking, an extractor is a function which extracts truly random bits from a weakly 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 [Tre98], who discovered that the Nisan–Wigderson pseudorandom generator [NW94], previously only used in a computational setting, could be used to construct extractors. Trevisan's extractor improves on most previous constructions and is optimal for certain settings of the parameters. However, when one wants to extract all (or most) of the randomness from the weakly random source, Trevisan's extractor performs poorly, in that a large number of truly random "catalyst" bits are needed. In this paper, we give an extractor which extracts *all* of the randomness from the weakly random source using fewer truly random bits than any previous construction. This is accomplished by improving the combinatorial construction underlying the Nisan–Wigderson generator used in Trevisan's construction. Applying a construction of Wigderson and Zuckerman [WZ95], we also obtain improved expander graphs and depth-2 superconcentrators.

Extractors. A distribution X on $\{0,1\}^n$ is said to have *min-entropy* k if for all $x \in \{0,1\}^n$, $\Pr[X = x] \leq 2^{-k}$. Think of this as saying that X has "k bits of randomness." A function $\textsc{Ext}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is called an (k, ε)*-extractor* if for every distribution X on $\{0,1\}^n$ of min-entropy k, the induced distribution $\textsc{Ext}(X, U_d)$ on $\{0,1\}^m$ has statistical difference at most ε from uniform (where U_d is the uniform distribution on $\{0,1\}^d$). In other words, $\textsc{Ext}$ extracts m (almost) truly random bits from a source with k bits of hidden randomness using d additional random bits as a catalyst. The goal is to explicitly construct extractors which minimize d (ideally, $d = O(\log(n/\varepsilon))$) while m is as close to k as possible.[1] *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 $\{0,1\}^m$.

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]. 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, TS98b, Tre98]. 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 [Tre98], 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]; derandomizing BPP under circuit-complexity assumptions [ACR97, STV98], and other problems in complexity theory [Sip88, GZ97].

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

Our results. The first two extractors constructed in this paper are given in the following theorem:

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

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

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

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

In particular, using the first extractor with k/m constant, we can extract any constant fraction of the source min-entropy using $O(\log^2 n)$ additional random bits, and, using the second extractor with $k = m$, we can extract all of the source min-entropy using $O((\log^2 n)(\log k))$ additional random bits. Theorem 1 has also been independently obtained by Ran Raz and Omer Reingold.

A comparison of these extractors with the best previous constructions is given in Figure 1 (for simplicity, we hide the dependence on ε, the statistical difference from uniform achieved). Our second extractor directly improves that of Ta-Shma [NT98], in that ours uses $O((\log^2 n)(\log k)) \leq O(\log^3 n)$ truly random bits in comparison to $O(\log^9 n)$.[2] Both of our extractors use more truly random bits than the extractors of [Zuc97, Tre98] and the disperser of [TS98b], but our extractors have the advantage that they work for any min-entropy (unlike [Zuc97]), extract all (or a constant fraction) of the min-entropy (unlike [Tre98]), and are extractors rather than dispersers (unlike [TS98b]). The disadvantage of the extractors of [GW97] described in Figure 1 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., $k = n - \text{polylog}(n)$), whereas ours uses $O(\log^3 n)$ random bits for any min-entropy. 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 entropy loss.

reference	min-entropy k	output length m	additional randomness d	type
[GW97]	any k	$m = k$	$d = O(n - k)$	extractor
[Zuc97]	$k = \Omega(n)$	$m = (1-\alpha)k$	$d = O(\log n)$	extractor
[NT98]	any k	$m = k$	$d = O(\log^9 n)$	extractor
[TS98b]	any k	$m = k - \text{polylog}(n)$	$d = O(\log n)$	disperser
[Tre98]	$k = n^{\Omega(1)}$	$m = k^{1-\alpha}$	$d = O(\log n)$	extractor
	any k	$m = k^{1-\alpha}$	$d = O((\log^2 n)/(\log k))$	extractor
this paper	any k	$m = (1-\alpha)k$	$d = O(\log^2 n)$	extractor
	any k	$m = k$	$d = O((\log^2 n)(\log k))$	extractor
ultimate goal	any k	$m = k$	$d = O(\log n)$	extractor

Above, α is an arbitrarily small constant.

Figure 1: Comparison with best previous constructions

Plugging our second extractor into a construction of [WZ95] immediately yields the following expander graphs:

Corollary 2 *For every N and $K \leq N$, there is an explicitly constructible 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 degree compares with a degree bound of $(N/K) \cdot 2^{O((\log\log N)^9)}$ due to Ta-Shma [NT98]. Such expanders have applications to sorting and selecting in rounds, constructing superconcentrators, and constructing non-blocking networks [Pip87, AKSS89, WZ95], and the improvements of Corollary 2 translate to similar improvements in each of these applications. We remark that the construction of [WZ95] used to obtain Corollary 2 requires extractors that extract nearly all the entropy of the source.

The Trevisan extractor. The main tool in the Trevisan extractor is the Nisan–Wigderson generator [NW94], which builds a pseudorandom generator out of any predicate P such that the security 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 $P\colon \{0,1\}^\ell \to \{0,1\}$ be any predicate. For a string

[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 [TS98a] estimates the degree of this polynomial to be 9.

$y \in \{0,1\}^d$, 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 $\mathrm{NW}_{\mathcal{S},P}\colon \{0,1\}^d \to \{0,1\}^m$ 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 $D\colon \{0,1\}^m \to \{0,1\}$ 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 $\textsc{Ext}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$. 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 (whose properties are unimportant in this informal description) and define $\ell = \log \overline{n}$. We view $\overline{u}$ as a boolean function $\overline{u}\colon \{0,1\}^\ell \to \{0,1\}$. 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 [Tre98] shows that the output of this extractor is close to uniform as long as the source min-entropy required is greater than the size of the circuit built in the security reduction of [NW94]. Hence, one needs to keep this circuit size small while minimizing the number d of truly random bits needed, which is equal to the seed length of the Nisan–Wigderson generator. Unfortunately, using $\max_{i \neq j} |S_i \cap S_j|$ as the measure of the circuit size as in [NW94, Tre98], one cannot make d much smaller than what is obtained in [Tre98].

The improvement. The improvements of this paper stem 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 security reduction. 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 with the corresponding designs. The weak designs used in our first extractor are constructed using an application of the Probabilistic Method, which we then derandomize using the Method of Conditional Expectations (see [ASE92] and [MR95, Ch. 5]). We then apply a simple iteration to these first weak designs to obtain the weak designs used in our second extractor. We also prove a lower bound showing that our weak designs are near-optimal.

Entropy loss. Since a (k, ε)-extractor $\textsc{Ext}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ 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 simulataneously minimize both d and the entropy loss.

Nonconstructively, one can show that, for any n and $k \leq n$, there exist extractors $\textsc{Ext}_{n,k}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{k+d-\Delta}$ with $d = \log(n-k) + O(1)$ and entropy loss $\Delta = 2\log(1/\varepsilon) + O(1)$, and these bounds on d and Δ are tight up to additive constants [RT97]. The explicit constructions, however, are still far from achieving these parameters. As for what is known, every entry in Figure 1 has an entropy loss of $d + m - k$, by definition. For example, the extractor of [GW97] has an entropy loss of $O(n-k)$ (which is only interesting when k is very close to n) and the extractor of [NT98] and the disperser of [TS98b] have entropy losses of $\mathrm{polylog}\, n$. In addition, the "tiny families of hash functions" of [SZ98] give extractors with $d = O(k + \log n)$ and entropy loss $2\log(1/\varepsilon) + O(1)$; these have optimal entropy loss but are only interesting when k is very small (e.g., $k = \mathrm{polylog}\, n$), as d is linear in k.

By combining our second extractor with one of [SZ98], we are able to achieve optimal entropy loss (up to an additive constant):

Theorem 3 *For every n, k, and ε such that $k \leq n$, there is an explicit (k, ε)-extractor* $\textsc{Ext}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{k+d-\Delta}$ *with*

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

and entropy loss

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

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 2.

reference	additional randomness d	entropy loss Δ	type
[GW97]	$d = O(n - k + \log(1/\varepsilon))$	$\Delta = n - k + 12\log(1/\varepsilon) + O(1)$	extractor
[SZ98]	$d = O(k + \log n)$	$\Delta = 2\log(1/\varepsilon) + O(1)$	extractor
[NT98]	$d = O((\log^9 n)(\log(1/\varepsilon)))$	$\Delta = O((\log^9 n)(\log(1/\varepsilon)))$	extractor
[TS98b]	$d = O(\log(n/\varepsilon))$	$\Delta = \mathrm{polylog}(n/\varepsilon)$	disperser
this paper	$d = O((\log^2(n/\varepsilon))(\log k))$	$\Delta = 2\log(1/\varepsilon) + O(1)$	extractor
nonconstructive & optimal [RT97]	$d = \log(n - k) + O(1)$	$\Delta = 2\log(1/\varepsilon) + O(1)$	extractor

All of the above work for any source of min-entropy k.
ε is the statistical difference from uniform achieved.

Figure 2: Results on entropy loss

Improved pseudorandom generators. Using a relaxed notion of designs also gives some quantitative improvements over [NW94] in the construction of pseudorandom generators from hard predicates. Details of these improvements are given in Section 8.

2 Preliminaries

In this section, we introduce some standard terminology and notation used throughout the paper. "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. 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") $D\colon \{0,1\}^m \to \{0,1\}$. A distribution X is said to have *min-entropy* k if for all x, $\Pr[X = x] \leq 2^{-k}$. It is useful to think of distributions of min-entropy k as being uniform over a subset of the domain of size 2^k.

We write U_j for the uniform distribution on strings of length j. A function $\textsc{Ext}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is a (k, ε)-*extractor* if for every distribution X of min-entropy k, $\textsc{Ext}(X, U_d)$ has statistical difference at most ε from U_m. We say that a family of extractors $\{\textsc{Ext}_i\colon \{0,1\}^{n_i} \times \{0,1\}^{d_i} \to \{0,1\}^{m_i}\}_{i \in I}$ is *explicit* if $\textsc{Ext}_i$ can be evaluated in time $\mathrm{poly}(n_i, d_i)$.

3 Combinatorial designs

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

Definition 4 ([NW94]) [3] *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 $i \neq j$, $|S_i \cap S_j| \leq \log \rho$.*

Motivation. In Trevisan's extractor, the parameters of a design correspond to the parameters of the extractor as follows:

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

Hence, our goal in constructing designs is to minimize d given parameters m, ℓ, and ρ (such that $\rho > 1$). 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 5 ([NW94, Tre98]) *For every m, ℓ, and $\rho > 1$, there exists an efficiently constructible (ℓ, ρ)-design $S_1, \ldots, S_m \subset [d]$ with*

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

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^c truly random bits for some $c > 0$. 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 6 *If $S_1, \ldots, S_m \subset [d]$ is an (ℓ, ρ)-design, then*

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

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

Definition 7 *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 8 *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 $\operatorname{poly}(m, d)$.

[3] 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.

This is already much better than what is given by Lemma 5; for constant ρ, d is $O(\ell^2)$ instead of $\ell^2 m^c$. 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 8 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 $\sum_{j<i} 2^{|S_i \cap S_j|} < \rho \cdot (i-1)$.[4] 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 8 (directly inspired by the iteration of Wigderson and Zuckerman [WZ95] on extractors), and yields the following improvement.

Lemma 9 *For every $\ell, m \in \mathbb{N}$ and $3/m \le \gamma < 1/2$, there exists a weak $(m, \ell, 1+\gamma, d)$-design with*

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

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

In particular, we can take $\gamma = \Theta(1/m)$ and extract essentially all of the entropy of the source using $d = O(\ell^2 \log m)$ truly random bits. Lemma 9 will be proven in Section 5.

For extractors which use only $O(\log n)$ truly random bits, where n is the input length, 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 10 *For every (ℓ, ρ)-weak design $S_1, \ldots, S_m \subset [d]$,*

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

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

4 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 [Tre98] 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 $P\colon \{0,1\}^\ell \to \{0,1\}$ 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:

Lemma 11 (Error-correcting codes) *For every n and δ there is a code $\mathrm{EC}_{n,\delta}\colon \{0,1\}^n \to \{0,1\}^{\overline{n}}$ where $\overline{n} = \mathrm{poly}(n, 1/\delta)$ such that every Hamming ball of relative radius $1/2 - \delta$ in $\{0,1\}^{\overline{n}}$ contains at most $1/\delta^2$ codewords. Furthermore, $\mathrm{EC}_{n,\delta}$ can be evaluated in time $\mathrm{poly}(n, 1/\delta)$ and $\overline{n}$ can be assumed to be a power of 2.*

We will actually use a stronger property of such error-correcting codes:

Proposition 12 *Let $\mathrm{EC}_{n,\delta}$ be any family of codes such that every Hamming ball of relative radius $1/2 - \delta$ contains fewer than B codewords and $\ln B \le \delta^2 \cdot \overline{n}$. Then every ℓ_1-ball of relative radius $1/2 - 2\delta$ in $[0,1]^{\overline{n}} \subset \mathbb{R}^{\overline{n}}$ contains at most B codewords. In other words, for every real vector $v \in [0,1]^{\overline{n}}$,*

$$\#\left\{u \;\middle|\; \frac{1}{\overline{n}} \sum_{i=1}^{\overline{n}} |\mathrm{EC}(u)_i - v_i| < \frac{1}{2} - 2\delta\right\} \le B$$

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

Since we may assume that $\overline{n} \geq \delta^{-2} \ln \delta^{-2}$ in Lemma 11, Proposition 12 applies to those codes with $B = 1/\delta^2$. Both Lemma 11 and Proposition 12 are justified in Appendix A.

We can now describe the Trevisan extractor, which takes as parameters n, m, k, and ε, where $m \leq n \leq k$. Let $\mathrm{EC}\colon \{0,1\}^n \to \{0,1\}^{\overline{n}}$ 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 $\overline{u}\colon \{0,1\}^\ell \to \{0,1\}$. Let $\mathcal{S} = (S_1, \ldots, S_m)$ be a collection of subsets of $[d]$ (for some d) such that $|S_i| = \ell$ for each i. (How $\mathcal{S}$ is selected will crucially affect the performance of the extractor; we will later choose it to be one of our weak designs.)

Then the extractor $\mathrm{E{\scriptstyle XT}}_{\mathcal{S}}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is defined as

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

We will now analyze this extractor. The following lemma is implicit in [NW94] and is more explicitly shown in [Tre98]. It shows how, from any function D which distinguishes the output of $\mathrm{NW}_{\mathcal{S},P}$ from uniform, one can obtain a (randomized) "program" which, using D as an oracle, predicts P with noticeable advantage. This lemma shows that this "program" can be taken to be of a very simple form, which will allow us to bound its complexity later on. The only modification we need to the proofs of [NW94, Tre98] is that we do not use an averaging argument to fix the "random part" of the hybrids in the "hybrid argument"; rather, we keep these random.

Lemma 13 *Let $P\colon \{0,1\}^\ell \to \{0,1\}$ be any predicate and let $D\colon \{0,1\}^m \to \{0,1\}$ be any "distinguisher" such that*

$$\Pr_r [D(r) = 1] - \Pr_y [D(\mathrm{NW}_{\mathcal{S},P}(y)) = 1] > \varepsilon,$$

where r is selected uniformly from $\{0,1\}^m$ and y from $\{0,1\}^d$. Then there exists a there exists an $i \leq m$ and functions $P_1, \ldots, P_i$ from $\{0,1\}^\ell$ to $\{0,1\}$

1. *$\Pr_{x,b,r} [D(P_1(x), \ldots, P_{i-1}(x), b, r) \oplus b = P(x)] > \frac{1}{2} + \frac{\varepsilon}{m}$, where x is selected uniformly from $\{0,1\}^\ell$, b from $\{0,1\}$, and r from $\{0,1\}^{m-i}$.*
2. *Each P_j depends on only $|S_i \cap S_j|$ bits of x (where these bit positions depend only on $\mathcal{S}$ and i, but not on P or D)*

Proof sketch: We can expand the hypothesis of Lemma 13 as

$$\Pr_{r_1 \cdots r_m} [D(r_1 \cdots r_m) = 1] - \Pr_y [D(P(y|_{S_1}) \cdots P(y|_{S_m})) = 1] > \varepsilon,$$

where $r_1, \ldots, r_m$ are uniformly and independently selected bits and y is uniformly selected from $\{0,1\}^d$. By the "hybrid argument" of [GM84] (cf. [Gol95, Sec. 3.2.3]), there is an i such that

$$\Pr_{y, r_i \cdots r_m} \left[D\left(P(y|_{S_1}) \cdots P(y|_{S_{i-1}}) r_i \cdots r_m\right) = 1\right] - \Pr_{y, r_{i+1} \cdots r_m} [D(P(y|_{S_1}) \cdots P(y|_{S_i}) r_{i+1} \cdots r_m) = 1] > \varepsilon/m$$

Now, renaming r_i as b and using the standard transformation from distinguishers to predictors [Yao82] (cf. [Gol98, Sec. 3.3.3]), we see that

$$\Pr_{y, b, r_{i+1} \cdots r_m} \left[D\left(P(y|_{S_1}) \cdots P(y|_{S_{i-1}}) b r_{i+1} \cdots r_m\right) \oplus b = P(y|_{S_i})\right] > \frac{1}{2} + \frac{\varepsilon}{m}$$

Using an averaging argument we can fix all the bits of y outside S_i while preserving the prediction advantage. 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

$$\Pr_{x, b, r_{i+1} \cdots r_m} [D(P_1(x) \cdots P_{i-1}(x) b r_{i+1} \cdots r_m) \oplus b = P(x)] > \frac{1}{2} + \frac{\varepsilon}{m},$$

as desired. □

Now we use a counting argument to bound the complexity (or "description size") of the "program" above and illustrate the connection with weak designs:

Lemma 14 *There is a set $\mathcal{F}$ of functions from $\{0,1\}^{\ell+1+m}$ to $\{0,1\}^m$ (depending only on $\mathcal{S}$) such that*

1. *For every predicate $P\colon \{0,1\}^\ell \to \{0,1\}$, distinguisher $D\colon \{0,1\}^m \to \{0,1\}$, and $\varepsilon > 0$ such that*

$$\Pr_r[D(r)=1] - \Pr_y[D(\mathrm{NW}_{\mathcal{S},P}(y))=1] > \varepsilon,$$

there exists a function $f \in \mathcal{F}$ such that

$$\Pr_{x,b,r}[D(f(x,b,r)) \oplus b = P(x)] > \frac{1}{2} + \frac{\varepsilon}{m},$$

where x is selected uniformly from $\{0,1\}^\ell$, b from $\{0,1\}$, and r from $\{0,1\}^m$.

2. $\log|\mathcal{F}| \le \log m + \max_i \left(\sum_{j<i} 2^{|S_i \cap S_j|}\right)$.

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

We will not use Item 3 (the bound on circuit size) in the analysis of our extractor; we only use this for our quantitative improvement to the pseudorandom generators of [NW94] given in Section 8.

Proof: By Lemma 13, to meet Condition 1 it suffices to let $\mathcal{F}$ be the set of functions f of the form $(x,b,r) \mapsto (P_1(x), P_2(x), \ldots, P_{i-1}(x), b, r)$, 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 i is $\log m$. Given i, 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 $\mathcal{F}$ is $\log m + \max_i \sum_{j<i} 2^{|S_i \cap S_j|}$, giving the desired bound on $\log|\mathcal{F}|$. 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 $\mathrm{Ext}_{\mathcal{S}}$ when we take $\mathcal{S}$ to be a weak design. The argument follows the analysis of Trevisan's extractor in [Tre98] except that we use the more refined bounds on $|\mathcal{F}|$ given by Lemma 14 and we use the stronger property of the error-correcting code given by Proposition 12 to reason about "randomized" approximations to the codewords $\overline{u}$.

Proposition 15 *If $\mathcal{S} = (S_1, \ldots, S_m)$ (with $S_i \subset [d]$) is a weak (ℓ,ρ)-design for $\rho = (k - 3\log(m/\varepsilon) - 5)/m$, then $\mathrm{Ext}_{\mathcal{S}}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is a (k,ε)-extractor.*

Proof: Let X be any distribution of min-entropy k. We need to show that the statistical difference between U_m and $\mathrm{Ext}(X, U_d)$ is at most ε. By the definition of statistical difference, this is equivalent to showing that for every distinguisher $D\colon \{0,1\}^m \to \{0,1\}$,

$$\Pr_r[D(r)=1] - \Pr_{u\leftarrow X,y}[D(\mathrm{NW}_{\mathcal{S},\overline{u}}(y))=1] \le \varepsilon$$

where r and y are selected uniformly from $\{0,1\}^m$ and $\{0,1\}^d$, respectively. (We have dropped the absolute value in the definition of statistical difference; this is without loss of generality since we may replace D by its binary complement.) So let $D\colon \{0,1\}^m \to \{0,1\}$ be any distinguisher and let $\mathcal{F}$ be as in Lemma 14, so $|\mathcal{F}| \le m2^{\rho m}$. For every $f \in \mathcal{F}$, we obtain a function $\hat{f}\colon \{0,1\}^\ell \to [0,1]$ given by

$$\hat{f}(x) = \Pr_{b,r}[D(f(x,b,r)) \oplus b = 1].$$

Think of $D(f(x,b,r)) \oplus b$ as a randomized algorithm built out of D with input x and random coins (b,r). We can view each $\hat{f}$ as a vector $\hat{f} \in [0,1]^{\overline{n}}$. Notice that for a predicate $P\colon \{0,1\}^\ell \to \{0,1\}$ (which we can also

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

view as a vector $P \in \{0,1\}^{\overline{n}}$), the ℓ_1-distance between $\hat{f}$ and P gives the probability that the randomized algorithm corresponding to f computes P incorrectly. That is,

$$|\hat{f} - P|_1 = \Pr_{b,r}[D(f(x,b,r)) \oplus b \neq P(x)].$$

Let B be the set of u for which there exists an $f \in \mathcal{F}$ such that $|\hat{f} - \overline{u}|_1 < 1/2 - \varepsilon/2m$. In other words, B is the set of "bad" u for which $\overline{u}$ can be approximated (with error $< 1/2 - \varepsilon/2m$) by one of these randomized algorithms $\hat{f}$. By the property of the error-correcting code given in Proposition 12, for each function $f \in \mathcal{F}$, there are at most $(4m/\varepsilon)^2$ strings $u \in \{0,1\}^n$ such that $|\hat{f} - \overline{u}|_1 < 1/2 - \varepsilon/2m$. By the union bound,

$$|B| \leq (4m/\varepsilon)^2 \cdot |\mathcal{F}| \leq (4m/\varepsilon)^2 \cdot m2^{\rho m}.$$

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

$$\begin{aligned}
\Pr_{u \leftarrow X}[u \in B] &\leq \left((4m/\varepsilon)^2 m 2^{\rho m}\right) \cdot 2^{-k} \\
&= \left((4m/\varepsilon)^2 m 2^{k - 3\log(m/\varepsilon) - 5}\right) \cdot 2^{-k} \\
&\leq \varepsilon/2
\end{aligned}$$

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

$$\Pr_r[D(r) = 1] - \Pr_y[D(\mathrm{NW}_{\mathcal{S},\overline{u}}(y)) = 1] \leq \varepsilon/2.$$

Thus,

$$\begin{aligned}
\Pr_r[D(r) = 1] - \Pr_{u \leftarrow X, y}[D(\mathrm{NW}_{\mathcal{S},\overline{u}}(y)) = 1] &= \mathop{\mathrm{E}}_{u \leftarrow X}\left[\Pr_r[D(r) = 1] - \Pr_y[D(\mathrm{NW}_{\mathcal{S},\overline{u}}(y)) = 1]\right] \\
&\leq \Pr_{u \leftarrow X}[u \in B] + \Pr_{u \leftarrow X}[u \notin B] \cdot \varepsilon/2 \\
&\leq \varepsilon/2 + \varepsilon/2 = \varepsilon.
\end{aligned}$$

■

Combining Proposition 15 with the weak designs given by Lemmas 8 and 9 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 $\Delta = 3\log(m/\varepsilon) + 5$ 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 $\Delta = 3\log(m/\varepsilon) + 5$. Let $k' = k - \Delta$, $m' = m - \Delta - 3$, 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 8 (resp., Lemma 9). This gives an (k, ε)-extractor $\textsc{Ext}\colon \{0,1\}^n \times \{0,1\}^{d'} \to \{0,1\}^{m'}$, 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 $\Delta + 3$ additional bits in the seed and simply concatenating these to the output, we obtain a (k, ε)-extractor $\textsc{Ext}\colon \{0,1\}^n \times \{0,1\}^{d'+\Delta+3} \to \{0,1\}^m$, as desired. (In applying Lemma 9, we need to make sure that $\rho < 3/2$, but if $\rho \geq 3/2$, we can use the weak design of Lemma 8 instead.) ■

Remark 16 A finer analysis can improve Proposition 15 to use $\rho = (k - 2\log(1/\varepsilon) - 3\log m - 6)/m$ rather than $\rho = (k - 3\log(m/\varepsilon) - 5)/m$. This is achieved by partitioning the set of "bad" u for which $\overline{u}$ can be approximated (by one of the randomized algorithms built out of D), into sets B_i according to the quality of approximation (e.g., take B_i to be those u for which $\overline{u}$ can be approximated with error between $1/2 - 2^i\varepsilon/2m$ and $1/2 - 2^{i-1}\varepsilon/2m$). Doing the analysis separately for each B_i has the effect of balancing the probability that $u \leftarrow X$ lands in B_i against the maximum advantage possible for u in B_i.

Remark 17 The stronger property of error-correcting codes given by Proposition 12 which corresponds to hardness against randomized algorithms could have been avoided by using an averaging argument to "fix" r and b in Lemma 13. If this is done in a straightforward manner, we would have to pay a price for these bits in the size of $\mathcal{F}$, as they would be needed to fully describe a function. The cost of these bits can be avoided, however, if we do the hybrid argument while we are still looking at the advantage of the distinguisher *averaged over the choice of* u; then these bits can be fixed independently of u and absorbed into the distinguisher *before* we make the counting argument which says that with high probability ($> 1 - \varepsilon/2$) over the choice of $u \leftarrow X$, $\overline{u}$ cannot be approximated (with error $< 1/2 - \varepsilon/2m$) by one of the (now deterministic) algorithms built out of D. Doing the analysis this way eliminates the $\log m$ term in the bound on $\log|\mathcal{F}|$ (since i has also been fixed independently of u). However, we feel that the approach we have taken is more straightforward.

5 Construction of weak designs

Proof of Lemma 8: 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 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 18 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 9: For simplicity, assume that $1+\gamma = 1/(1-2^{-h})$ and $m = 2^q/(1+\gamma)$. Let $d_0 = \lceil \ell/\ln 2 \rceil \cdot \ell$ and let $d = h \cdot d_0 = O(\ell^2 \cdot \log(1+\gamma))$. We view $[d]$ as the disjoint union of h blocks $B_1, \ldots, B_h$ each of size d_0. For each $t \in [h]$, let $m_t = 2^{q-t}$ and $n_t = \sum_{s=1}^{t-1} m_s$, so $\sum_t m_t = m$.

Now we define our weak design $S_1, \ldots, S_m$. For each $t \in [h]$, 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 8. 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 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

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

as desired. ■

6 Lower bounds for designs

Proof of Proposition 6: 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 10: We have

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

where the last inequality follows from Jensen's inequality. Thus,

$$\log 2\rho > \frac{1}{m^2} \sum_{i \neq j} |S_i \cap S_j| \qquad (1)$$

Now, for $a \in [d]$, let $n_a = |\{i{:}\, a \in S_i\}|$. Then $\sum_a n_a = \sum_i |S_i| = m \cdot \ell$.

$$\begin{aligned}
\sum_{i \neq j} |S_i \cap S_j| &= \sum_{a \in [d]} n_a(n_a - 1) \\
&= \sum_a n_a^2 - \sum_a n_a \\
&= \sum_a n_a^2 - m\ell \\
&\geq \frac{1}{d}\left(\sum_a n_a\right)^2 - m\ell \\
&= \frac{m^2\ell^2}{d} - m\ell \\
&\geq \frac{m^2\ell^2}{2d},
\end{aligned}$$

unless $d \geq (m\ell)/2$. Putting this in Inequality 1, we have

$$\log 2\rho > \frac{1}{m^2} \cdot \frac{m^2\ell^2}{2d} = \frac{\ell^2}{2d}$$

which proves the proposition. ■

Remark 19 The above proof gives a stronger bound on d if we have a family of sets $S_1, \ldots, S_m$ such that for all i, $\sum_{j<i} 2^{|S_i \cap S_j|} < \rho \cdot (i-1)$ (e.g., the family of sets constructed in the proof of Lemma 8). If we have such a bound, then summing over i from 1 to m gives

$$\rho \cdot \binom{m}{2} > \frac{1}{2} \sum_{i \neq j} 2^{|S_i \cap S_j|},$$

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

$$\log \rho > \frac{1}{m^2} \sum_{i \neq j} |S_i \cap S_j|$$

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

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

7 Achieving optimal entropy loss

Recall that the *entropy loss* of an extractor $\text{Ext}{:}\, \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is defined as $\Delta = k + d - m$, and we can hope for this to be as small as $2\log(1/\varepsilon) + O(1)$ with $d = \log(n-k) + O(1)$ [RT97].

In constructing our extractor $\text{Ext}_{\mathcal{S}}(u,y) = \text{NW}_{\mathcal{S},\overline{u}}(y)$, we "threw away" y after using it as a seed for the Nisan–Wigderson generator and hence the d bits of entropy carried by y were lost. However, the analysis of the Nisan–Wigderson generator actually shows that the quality of the generator is not affected if the seed is revealed. Thus, we define $\text{Ext}'_{\mathcal{S}}(u,y) = (y, \text{NW}_{\mathcal{S},\overline{u}}(y))$. Now all the analysis of Ext done in Section 4 actually applies to Ext′ (in Lemmas 13 and 14, give the distinguisher D the seed y in addition to $\text{NW}_{\mathcal{S},P}(y)$), and we obtain the following strengthening of Proposition 15:

Proposition 20 *If $\mathcal{S} = (S_1, \ldots, S_m)$ (with $S_i \subset [d]$) is a weak (ℓ,ρ)-design for $\rho = (k - 3\log(m/\varepsilon) - 5)/m$, then $\text{Ext}'_{\mathcal{S}}{:}\, \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{m+d}$ is a (k,ε)-extractor.*

Combining Proposition 20 and Lemma 9 with $m = k - 3\log(k/\varepsilon) - 6$ immediately gives us logarithmic entropy loss:

Proposition 21 *For every n, k, and ε such that $k \leq n$, there is an explicit (k,ε)-extractor* $\textsc{Ext}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{k+d-\Delta}$ *with*

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

and entropy loss

$$\Delta = 3\log(k/\varepsilon) + O(1)$$

By following Remarks 16 and 17, the entropy loss can be reduced to $2\log(k/\varepsilon) + O(1)$, but we will use a different method to make the entropy loss even better. We use a slight modification of an idea due to Wigderson and Zuckerman [WZ95]: Suppose we have a (k,ε)-extractor $\textsc{Ext}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{k+d-\Delta}$ with entropy loss Δ. Now, if x is taken from a source of min-entropy k and y from the uniform distribution on $\{0,1]\}^d$, then conditioned on "most" values of $\textsc{Ext}(X, U_d)$, the pair (x,y) 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 strengthens the one in [WZ95]:

Lemma 22 *Let $s > 0$. Suppose* $\textsc{Ext}_1\colon \{0,1\}^n \times \{0,1\}^{d_1} \to \{0,1\}^{m_1}$ *is a (k,ε_1)-extractor with entropy loss Δ_1 and* $\textsc{Ext}_2\colon \{0,1\}^{n+d_1} \times \{0,1\}^{d_2} \to \{0,1\}^{m_2}$ *is a $(\Delta_1 - s, \varepsilon_2)$-extractor with entropy loss Δ_2. Define* $\textsc{Ext}\colon \{0,1\}^n \times \{0,1\}^{d_1+d_2} \to \{0,1\}^{m_1+m_2}$ *by*

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

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

Lemma 22 has two main differences from the one in [WZ95]: First, we use the second extractor on the pair (x, y_1) rather than just x; this enables us to make the output length close to $k + d_1 + d_2$ rather than just k. Second, the statistical difference from uniform in $\textsc{Ext}$ has a better dependence on s (in [WZ95], the expression is $\varepsilon_1 + \varepsilon_2 + 2^{-s}$). Before proving Lemma 22, let us see how we can use it to make our entropy loss optimal. If we use the extractor given by Proposition 21 as $\textsc{Ext}_1$, Lemma 22 tells us that we need only find an extractor $\textsc{Ext}_2$ which works well for very small (i.e., logarithmic) min-entropy. The following extractor of Srinivasan and Zuckerman [SZ98] achieves exactly what we want:

Lemma 23 ([SZ98]) *For every n, $k \leq n$, and $\varepsilon > 0$, there is an explicit (k,ε)-extractor* $\textsc{Ext}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{k+d-\Delta}$ *with entropy loss $\Delta = 2\lceil\log(1/\varepsilon)\rceil + 2$ and $d = O(k + \log n)$.*

Now we prove Theorem 3:

Proof of Theorem 3: Let n, $k \leq n$, and $\varepsilon > 0$ be given. By Proposition 21, there is an explicit $(k, \varepsilon/4)$-extractor $\textsc{Ext}_1\colon \{0,1\}^n \times \{0,1\}^{d_1} \to \{0,1\}^{m_1}$ with $d_1 = O((\log^2(n/\varepsilon))(\log k))$ and entropy loss $\Delta_1 = O(\log(m/\varepsilon))$. By Lemma 23, there is an explicit $(\Delta_1 - 1, \varepsilon/2)$-extractor $\textsc{Ext}_2\colon \{0,1\}^{n+d_1} \times \{0,1\}^{d_2} \to \{0,1\}^{m_2}$ with $d_2 = O(\Delta_1 + \log(n+d_1)) = O(\log(n/\varepsilon))$ 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 22 (with $s = 1$), there is an explicit (k,ε)-extractor $\textsc{Ext}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ with $d = d_1 + d_2 = O((\log^2(n/\varepsilon))(\log k))$ and entropy loss $\Delta = \Delta_2 + 1 = 2\lceil\log(1/\varepsilon)\rceil + 5$. ■

Remark 24 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 Remarks 16 and 17, we obtain a version of Proposition 20 in which $\rho = (k - 2\log(m/\varepsilon) - O(1))/m$. In particular, we have for any $k > 2\log(1/\varepsilon) + O(1)$, a (k,ε)-extractor $\textsc{Ext}^*\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{d+2}$ with $d = O(\log^2(n/\varepsilon))$. We can repeatedly use this extractor (with careful settings of k and ε) as $\textsc{Ext}_2$ in Lemma 22 to reduce the entropy loss of the extractor given by Proposition 21 (i.e., we start with the extractor of Proposition 21 as $\textsc{Ext}_1$ and obtain an extractor with smaller entropy loss which becomes $\textsc{Ext}_1$ in the next application of Proposition 21). The fact that $\textsc{Ext}^*$ 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 22: Let X be any source of min-entropy k. Let L be the set of strings $z \in \{0,1\}^{m_1}$ such that $\Pr[\textsc{Ext}_1(X, U_{d_1}) = z] < 2^{-(m_1+s)}$ (i.e., L is the set of "light" strings in the output of $\textsc{Ext}_1$ — strings that occur with probability that is smaller than uniform by a factor of 2^s). Recall that the entropy loss is defined as $\Delta_1 = k + d_1 - m_1$.

Claim 25 *For every $z \notin L$, the marginal distribution of (X, U_{d_1}) conditioned on $\textsc{Ext}_1(X, U_{d_1}) = z$ has min-entropy at least $\Delta_1 - s$.*

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

$$\begin{aligned} \Pr[(X, U_{d_1}) = (x, y) | \textsc{Ext}_1(X, U_{d_1}) = z] &= \frac{\Pr[(X, U_{d_1}) = (x, y)]}{\Pr[\textsc{Ext}_1(X, U_{d_1}) = z]} \\ &\leq \frac{2^{-k} \cdot 2^{-d_1}}{2^{-m_1-s}} \\ &= 2^{-(\Delta_1 - s)}. \end{aligned}$$

This proves Claim 25. □

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

Claim 26 $\Pr[\textsc{Ext}_1(X, U_{d_1}) \in L] \leq \varepsilon_1/(2^s - 1)$.

Proof of claim: For every $z \in L$, $\Pr[U_{m_1} = z] > 2^s \cdot \Pr[\textsc{Ext}_1(X, U_{d_1}) = z]$. Thus, $\Pr[U_{m_1} \in L] > 2^s \cdot \Pr[\textsc{Ext}_1(X, U_{d_1}) \in L]$. Now, by the definition of statistical difference,

$$\begin{aligned} \|U_{m_1} - \textsc{Ext}_1(X, U_{d_1})\| &\geq \Pr[U_{m_1} \in L] - \Pr[\textsc{Ext}_1(X, U_{d_1}) \in L] \\ &\geq (2^s - 1) \cdot \Pr[\textsc{Ext}_1(X, U_{d_1}) \in L] \end{aligned}$$

Since $\textsc{Ext}_1$ is a (k, ε_1)-extractor, this statistical difference is at most ε_1, and the claim follows. □

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

1. A has statistical difference at most ε_1 from U_{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_{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 $U_{m_1} \times U_{m_2} = U_{m_1+m_2}$, proving Lemma 22. ■

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 circuit complexity-based pseudorandom generation is the following:

Definition 27 *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 8, we obtain

Lemma 28 *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 $\mathrm{poly}(m, d)$.

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

Lemma 29 *Suppose $P: \{0,1\}^\ell \to \{0,1\}$ 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 28 with $s = 2m$ and ρ a small constant, we obtain

Theorem 30 *Suppose $P: \{0,1\}^\ell \to \{0,1\}$ 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 $G_{P,m}: \{0,1\}^{O(\ell^2/\log \ell)} \to \{0,1\}^m$ 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 $O(\ell^2)$).

Acknowledgments

I am most grateful to Luca Trevisan for sharing his novel insights into these problems with me. I thank Oded Goldreich for his neverending supply of guidance and support, and for all his help with the presentation. While doing this work, I had many useful discussions with Luca, Oded, and several other people — including Shafi Goldwasser, Adam Klivans, Madhu Sudan, Amnon Ta-Shma, and David Zuckerman. Some of these discussions yielded specific improvements in this paper, which I outline below:

The lower bound for (normal) designs in Proposition 6 was obtained in joint work with Luca Trevisan. The stronger property of error-correcting codes given by Proposition 12 was proved with the help of Madhu Sudan. Oded Goldreich provided the finer analysis of the extractor mentioned in Remark 16 and also advocated the more straightforward proof of Proposition 15 (considering randomized algorithms) instead of the alternative approach of Remark 17. The cleaner proof of Lemma 8 given in Section 5 was suggested by David Zuckerman (to replace a more complicated proof along the lines of Remark 18).

References

[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.

[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.

[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.

[GM84] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, April 1984.

[Gol95] Oded Goldreich. *Foundations of Cryptography (Fragments of a Book).* Weizmann Institute of Science, 1995. Available, along with revised version 1/98, from `http://theory.lcs.mit.edu/~oded`.

[Gol98] Oded Goldreich. *Modern Cryptography, Probabilistic Proofs and Pseudorandomness*, June 1998. To be published by *Springer.*

[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.

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

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

[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 [TS96].

[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.

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

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

[STV98] Madhu Sudan, Luca Trevisan, and Salil Vadhan. Pseudorandom generators without the XOR lemma. Unpublished manuscript, October 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.

[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*.

[Tre98] Luca Trevisan. Constructions of near-optimal extractors using pseudo-random generators. *Electronic Colloquium on Computational Complexity* Technical Report TR98-55, September 1998. `http://www.eccc.uni-trier.de/eccc`.

[TS96] 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.

[TS98a] Amnon Ta-Shma. Personal communication, August 1998.

[TS98b] 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.

[Vad98] Salil P. Vadhan. Extracting all the randomness from a weakly random source. *Electronic Colloquium on Computational Complexity* Technical Report TR98-47, August 1998. `http://www.eccc.uni-trier.de/eccc`.

[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 11: 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 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 Had(x) is the inner-product of x and y mod 2. Thus, for $x_1 \neq x_2$, Had(x_1) and Had(x_2) have (relative) Hamming distance 1/2. Let $F = \mathrm{GF}(2^m)$; an explicit description of F can be found in time $\mathrm{poly}(2^m) = \mathrm{poly}(n, 1/\delta)$ by exhaustively searching for an irreducible polynomial of degree m over GF(2). We can view strings $x \in \{0,1\}^n \subset (\{0,1\}^m)^{\lceil n/m \rceil}$ as giving the coefficients of a polynomial p_x of degree at most $d = \lceil n/m \rceil$ over F.

Now we define the error-correcting code 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 $p_x(a) \neq p_y(a)$, Had$(p_x(a))$ and Had$(p_y(a))$ disagree in $2^m/2$ positions. Thus, for distinct x and y, EC(x) and 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 31 *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. ■

Proof of Proposition 12: Suppose not, *i.e.* there are at least B codewords in an ℓ_1-ball of radius $1/2 - 2\delta$ around some real vector v. Then, consider a vector $w \in \{0,1\}^{\overline{n}}$ obtained by randomly rounding v. That is, let $w_i = 1$ with probability v_i for every i independently. Now consider any codeword $\overline{u}$ within ℓ_1-distance $1/2 - 2\delta$ of v. The Hamming distance between $\overline{u}$ and w has expectation $\|\overline{u} - v\|_1$ and is the sum of $\overline{n}$ independent $\{0,1\}$-valued random variables. By the Hoeffding inequality, the probability that the Hamming distance between $\overline{u}$ and w exceeds $\|\overline{u} - v\|_1 + \delta$ is at most $\exp(-2\overline{n}\delta^2)) = 1/|B|^2$. Hence, the probability that there are fewer than B codewords within Hamming distance $1/2 - \delta$ around w is at most $1/|B|$. Thus, there exists a w with at least B codewords within Hamming distance $1/2 - \delta$ contradicting the property of the error-correcting code. ■

B Derandomizing the proof of Lemma 8

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

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

for any i and $\alpha_1, \ldots, \alpha_i$. If we let $T = \{\alpha_1, \ldots, \alpha_i\}$, then a calculation like the one in the proof of Lemma 8 for the unconditional expectation shows

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

which can be easily computed.

ECCC ISSN 1433-8092
http://www.eccc.uni-trier.de/eccc
ftp://ftp.eccc.uni-trier.de/pub/eccc
ftpmail@ftp.eccc.uni-trier.de, subject 'help eccc'