Electronic Colloquium on Computational Complexity, Revision 2 of Report No. 106 (2020)

Explicit Extremal Designs and Applications to Extractors

Eshan Chattopadhyay[*]
Cornell University
eshanc@cornell.edu

Jesse Goodman[*]
Cornell University
jpmgoodman@cs.cornell.edu

July 15, 2020

Abstract

An (n, r, s)-design, or (n, r, s)-partial Steiner system, is an r-uniform hypergraph over n vertices with pairwise hyperedge intersections of size $< s$. An independent set in a hypergraph G is a subset of vertices covering no hyperedge, and its independence number $\alpha(G)$ is the size of its largest independent set. For all constants $r \geq s \in \mathbb{N}$ with r even, we explicitly construct (n, r, s)-designs $(G_n)_{n \in \mathbb{N}}$ with independence number $\alpha(G_n) \leq O(n^{\frac{2(r-s)}{r}})$. This gives the first derandomization of a result by Rödl and Šinajová (Random Structures & Algorithms, 1994).

By combining our designs with a recent explicit construction of a *leakage-resilient extractor* that works for low-entropy (Chattopadhyay et al., FOCS 2020), we obtain simple and significantly improved low-error explicit extractors for adversarial and small-space sources. In particular, for any constant $\delta > 0$, we extract from (N, K, n, k)-adversarial sources of locality 0, where $K \geq N^{\delta}$ and $k \geq \text{polylog}\, n$. The previous best result (Chattopadhyay et al., STOC 2020) required $K \geq N^{1/2+o(1)}$. As a result, we get extractors for small-space sources over n bits with entropy requirement $k \geq n^{1/2+\delta}$, whereas the previous best result (Chattopadhyay et al., STOC 2020) required $k \geq n^{2/3+\delta}$.

[*]Supported by NSF grant CCF-1849899.

ISSN 1433-8092

1 Introduction

Let $G = (V, E)$ be an r-uniform hypergraph over n vertices. G is called an (n, r, s)-*design*, or (n, r, s)-*partial Steiner system*, if $|e_1 \cap e_2| < s$, for all distinct $e_1, e_2 \in E$. In this paper, we study (n, r, s)-designs that are extremal in the sense that they have small independence number.[1] In 1994, Rödl and Šinajová proved the first general theorem about the existence of such objects:

Theorem 1.1 ([RŠ94]). *Given any $n \geq r \geq s \in \mathbb{N}$ with $r \geq 2$, there exists an (n, r, s)-design G with independence number*

$$\alpha(G) \leq C_{r,s} \cdot n^{\frac{r-s}{r-1}} (\log n)^{\frac{1}{r-1}},$$

where $C_{r,s} = C(r, s)$ depends only on r, s.

In fact, they also showed this result is tight up to the term $C_{r,s}$ that depends only on r, s.

In order to prove Theorem 1.1, Rödl and Šinajová used the Lovász Local Lemma [Spe77] to show that a random r-uniform hypergraph, where each hyperedge is included independently with some probability p, is an (n, r, s)-design with small independence number. Thus, while their result proves the existence of these extremal designs, it does not provide an explicit way to construct them. Furthermore, later work has focused on improving the term $C_{r,s}$ [EV13, Eus13] or extending the result to more general types of designs [GPR95, KMV14, TL18], while still relying on probabilistic constructions.

To the best of our knowledge, there are no known explicit constructions of (n, r, s)-designs with a nontrivial upper bound on their independence number. However, such constructions are important in both extremal combinatorics and theoretical computer science: not only do they offer insight into the structure of these objects, but they also have recently found applications in the construction of *randomness extractors* [CGGL20]. Furthermore, explicit designs of a different extremal flavor have been used to construct *pseudorandom generators* in the seminal work of Nisan and Wigderson [NW94].[2]

Thus, the primary goal - and first main contribution - of this paper is a derandomization of Theorem 1.1. Second (and third), we show that our explicit designs are strong enough to be used, along with a very recent object from [CGG+20], to establish an explicit framework for randomness extraction that greatly improves the results in [CGGL20]. Before we formally state our main theorems, we provide a brief refresher on randomness extractors.

Randomness extractors A *randomness extractor* is an object that can *purify* defective sources (distributions) of randomness found in nature. It is motivated by the fact that most applications of randomness in computing require access to uniformly random bits, yet the random bits harvested from natural phenomena rarely look so pure. An extractor is formally defined as follows.

Definition 1.2. *Let $\mathcal{X}$ be a family of sources over $\{0,1\}^n$. A function $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^m$ is an* extractor *for $\mathcal{X}$ with error ϵ if for every $\mathbf{X} \in \mathcal{X}$,*

$$|\mathsf{Ext}(\mathbf{X}) - \mathbf{U}_m| \leq \epsilon,$$

where $\mathbf{U}_m$ is the uniform distribution over $\{0,1\}^m$, and $|\cdot|$ denotes total variational distance.

[1] Recall that an *independent set* in a hypergraph is a subset of vertices that contains no edge, and the *independence number* of a hypergraph is the size of its largest independent set.

[2] [NW94] uses (n, r, s)-designs that are extremal in the sense that they have a large number of hyperedges. It is known how to explicitly construct this type of extremal design; see, e.g., the survey [Vad12].

Beyond purifying natural sources of randomness, extractors have found important applications in complexity theory, cryptography, coding theory, and combinatorics (see, e.g., [Sha11, Vad12]). Constructing these objects have thus produced a fruitful line of research over the past 30 years, where various source families $\mathcal{X}$ and errors ϵ have been considered depending on the motivating application. Our designs will give much improved extractors for two important classes $\mathcal{X}$ of sources.

First, we will give improved extractors for the class $\mathcal{X}$ of *adversarial sources.* Motivated by applications in generating a (cryptographic) common random string, and in harvesting randomness from unreliable sources, Chattopadhyay, Goodman, Goyal and Li [CGGL20] introduced this new class of sources:

Definition 1.3. *A source* $\mathbf{X}$ *over* $(\{0,1\}^n)^N$ *is an* (N, K, n, k)-adversarial source of locality 0 *if it has the form* $\mathbf{X} = (\mathbf{X}_1, \ldots, \mathbf{X}_N)$*, where each* $\mathbf{X}_i$ *is an independent source over* $\{0,1\}^n$*, and there exists some (unknown) subset of* K good sources*; i.e., there is some* $S \subseteq [N], |S| \geq K$ *such that each* $X_j, j \in S$ *has min-entropy*[3] $H_\infty(\mathbf{X}) \geq k$*.*

The class $\mathcal{X}$ of adversarial sources generalizes several well-studied models, which will allow us to obtain improved extractors of another type, discussed next.

Second, we will give improved extractors for the class $\mathcal{X}$ of *small space sources.* Introduced by Kamp, Rao, Vadhan and Zuckerman [KRVZ06], each distribution in this class is defined to be samplable from some algorithm with limited memory. To define this class of sources formally, one first defines a *branching program of width* w *and length* n to be a directed acyclic graph with $n+1$ layers, where the first layer (*layer 0*) has 1 node, the remaining layers have w nodes each, and such that an edge starting in layer i must terminate in layer $i+1$. Then, one formally defines a small space source as follows.

Definition 1.4. *A distribution* $\mathbf{X}$ *over* $\{0,1\}^n$ *is a* space s source *if it is generated by a random walk starting on the first layer of a branching program of width* 2^s *and length* n*, where each edge is labeled with an output bit and some transition probability.*

These sources were defined following a line of work introduced by Trevisan and Vadhan [TV00] on extracting uniform bits from distributions that can be sampled by algorithms with some limited computational resource. The motivation behind sources of this type is the idea that since the universe has limited resources, it may be reasonable to assume that imperfect sources of randomness found in nature are generated in this way.

Given these definitions, we are now ready to state our main contributions.

1.1 Summary of results

In our first main result, we explicitly construct (n, r, s)-designs with small independence number. In particular, we prove the following theorem.

Theorem 1. *There exists an Algorithm* $\mathcal{A}$ *such that given any* $n \geq r \geq s \in \mathbb{N}$ *as input with* r *an even number,* $\mathcal{A}$ *runs in time* $\text{poly}\left(\binom{n}{r}\right)$ *and outputs an* (n, r, s)*-design* G *with independence number*

$$\alpha(G) \leq C_{r,s} \cdot n^{\frac{2(r-s)}{r}},$$

where $C_{r,s} = C(r,s)$ *depends only on* r, s*.*

[3] The min-entropy of a source $\mathbf{X}$ over $\{0,1\}^n$ is defined as $H_\infty(\mathbf{X}) := \min_{x \in \text{support}(\mathbf{X})} \log(1/\Pr[\mathbf{X} = x])$.

For all constants $r \geq s \in \mathbb{N}$ where r is even, Theorem 1 gives an explicit family of (n, r, s)-designs $(G_n)_{n\in\mathbb{N}}$ with small independence number. The independence number of our explicit designs differs from Theorem 1.1 (known to be optimal, up to constants) by just a quadratic factor. Like the non-explicit designs of [RŠ94], our derandomization focuses on the case where r, s are constant. However, we also give meaningful results when r, s are not constant: we show that we may take $C_{r,s} = Cr^4$ for some universal constant $C > 0$, and we argue that for almost all interesting settings of *super-constant* r, s, our algorithm still runs efficiently. We refer the reader to Remark 3.2 in Section 3, and the discussion thereafter, for more detail.

Our second main theorem gives much improved explicit extractors for (N, K, n, k)-adversarial sources of locality 0:

Theorem 2. *There is a universal constant $C > 0$ such that for any fixed $\delta > 0$ and all sufficiently large $N, K, n, k \in \mathbb{N}$ satisfying $k \geq \log^C n$ and $K \geq N^\delta$, there exists an explicit extractor $\mathsf{Ext} : (\{0,1\}^n)^N \to \{0,1\}^m$ for (N, K, n, k)-adversarial sources of locality 0, with output length $m = k^{\Omega(1)}$ and error $\epsilon = 2^{-k^{\Omega(1)}}$.*

Previously, the best extractor for this setting [CGGL20] required $K \geq N^{0.5+o(1)}$ good sources, and was quite involved. We believe our extractor to be much simpler, and furthermore our construction gives a new application of *leakage-resilient extractors* (LREs). The variant of LREs that we use here were studied in a recent line of work [KMS19, CGG$^+$20] and are known as *extractors for cylinder intersections*. Most relevant to us is the work of Chattopadhyay et al. [CGG$^+$20] where they gave the first constructions of such objects for entropy $k = o(n)$, and in fact their LREs work for entropy $k \geq \operatorname{polylog} n$. Theorem 2 relies heavily on these low-entropy LREs.

Finally, our third main theorem gives new low-error explicit extractors for space s sources. Until recently, the best extractors for such sources [KRVZ06] required entropy $k \geq Cn^{1-\gamma}s^\gamma$, where $\gamma > 0$ is some tiny constant and C is a large one. In [CGGL20], the entropy requirement was improved to $k \geq Cn^{2/3+\delta}s^{1/3-\delta}$. We reduce this entropy requirement even further, and prove the following.

Theorem 3. *For any fixed $\delta \in (0, 1/2]$ there is a constant $C > 0$ such that for all $n, k, s \in \mathbb{N}$ satisfying $k \geq Cn^{1/2+\delta}s^{1/2-\delta}$, there exists an explicit extractor $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^m$ for space s sources of min-entropy k, with output length $m = n^{\Omega(1)}$ and error $\epsilon = 2^{-n^{\Omega(1)}}$.*

The line of improvements described above (from [KRVZ06] to [CGGL20] to Theorem 3) is *strict*, in terms of both entropy *and* space: we refer the reader to Remark 5.2 in Section 5 for more detail. Furthermore, the entropy requirement in Theorem 3 virtually reaches the known limit of current techniques in low-error extraction from small-space sources, and any further improvements (i.e., beyond $k \geq n^{1/2}$) will require significantly new ideas (see Remark 5.7 in Section 5). Finally, we note that, along the way, we provide improved randomness extractors for a class of sources known as *total entropy sources* (see Theorem 5.6).

1.2 Overview

We briefly sketch an overview of our techniques. We refer the reader to Section 2 for some basic preliminaries on notation and probability.

Explicit designs with small independence number In Section 3, we give our explicit construction of designs with small independence number, proving Theorem 1. In order to construct our (n, r, s)-designs $G = (V, E)$, we start with a linear code $Q \subseteq \mathbb{F}_2^n$ of distance $d > 2(r - s)$, and then restrict it to the set $Q_r \subseteq Q$ of elements in Q that have Hamming weight r. Our design

$G = (V, E)$ is constructed by identifying V with $[n]$, and by creating a hyperedge for each $x \in Q_r$ in the natural way. The distance of the code and the definition of Q_r immediately guarantees that G is an (n, r, s)-design.

In order to upper bound the independence number $\alpha(G)$ of our design, we observe that any independent set in G corresponds to a subcube $S \subseteq \mathbb{F}_2^n$ that contains no vector in Q of weight r; in other words, since Q is a *linear* code, this means that the *subspace* $T^* := S \cap Q$ has no vector of Hamming weight r. If our linear code Q had very high dimension, then even if the subcube S was relatively small, we would have found a relatively large subspace T^* containing no vector of Hamming weight r. But intuitively, it seems like as the dimension of a subspace grows large enough, at some point it must be guaranteed to have such a vector. It turns out this is true, and it follows immediately from Sidorenko's recent bounds [Sid18, Sid20] on the size of sets in $\mathbb{F}_2^n$ containing no r elements that sum to zero. Thus if Q has large enough dimension, S cannot be too large, and thus neither can $\alpha(G)$. All that remains is to explicitly construct (the weight-r vectors of) a high-dimensional linear code $Q \subseteq \mathbb{F}_2^n$ with distance $d > 2(r - s)$, which can easily be done using BCH codes [BRC60, Hoc59].

Improved extractor for adversarial sources In Section 4, we show how to combine our designs with leakage-resilient extractors (LREs) in order to obtain better extractors for adversarial sources, proving Theorem 2. Informally, an LRE for r sources offers the guarantee that its output looks uniform *even conditioned on* the output of many *leakage* functions, each called on up to $r - 2$ of the same inputs fed to the original LRE. We now sketch our extractor construction for adversarial sources. Recall that an (N, K, n, k)-adversarial source of locality 0 consists of N independent sources, where only K of them are guaranteed to be "good" (i.e., contain some entropy). Given such a source, we use Theorem 1 to explicitly construct an $(N, r, r-1)$-design G with small independence number $\alpha(G) < K$, and we identify the vertices of our design with the N independent sources. Then, for each hyperedge in our design, we call a leakage-resilient extractor on the r sources it contains, and finish by taking the bitwise XOR over the outputs of the LRE calls.

This construction successfully outputs uniform bits for the following reasons. Because $\alpha(G) < K$, we are guaranteed that *some* LRE call is given *only* good sources. By the *extractor* property of the LRE, this call will output uniform bits. Meanwhile, the *bounded intersection* property of the $(N, r, r - 1)$-design, paired with the *leakage-resilience* property of the LRE, guarantees that these uniform bits still look uniform *even after taking their bitwise XOR with the outputs of all other LRE calls.* Using these ideas, we actually provide a slightly more general framework to combine (N, r, s)-designs with LREs of various strength. Our framework leverages the "*activation vs. fragile correlation*" paradigm introduced in [CGGL20], yet it is able to do so in a much more simple, general, and effective way, by combining two very recent explicit pseudorandom objects: LREs from [CGG+20], and explicit designs from the current paper.

Improved extractor for small-space sources In Section 5, we show how our adversarial source extractors give better extractors for total-entropy sources and small-space sources, and thereby prove Theorem 3. To do so, we simply import standard reductions from small-space sources to total-entropy sources [KRVZ06] to adversarial sources [CGGL20]. The results then follow immediately. This application of adversarial source extractors was first provided in [CGGL20], but we include the (slightly optimized) techniques here for completeness, i.e., to demonstrate how the parameters of our improved adversarial source extractors carry over to total-entropy and small-space sources.

We conclude with some remarks and present some open problems in Section 6.

2 Preliminaries

General notation Given two strings $x, y \in \{0,1\}^m$, we let $x \oplus y$ denote their bitwise XOR. For a number $n \in \mathbb{N}$, $[n]$ denotes the interval $[1, n] \subseteq \mathbb{N}$. We let $\circ$ denote string concatenation, and for a collection $\{x_i : i \in I\}$ indexed by some finite set I, we let $(x_i)_{i \in I}$ denote the concatenation of all strings $x_i, i \in I$. If I is already equipped with some total order, this is used to determine the concatenation order; otherwise, I is arbitrarily identified with $[|I|]$ to induce a total ordering. Given a domain $\mathcal{D}$, and some string $x \in \mathcal{D}^N$, we let $x_i \in \mathcal{D}$ denote the value at the i^{th} coordinate of x. Given a subset $S \subseteq [N]$, we let $x_S := (x_i)_{i \in S}$. Even if $\mathcal{D} = \mathcal{R}^n$ for some other domain $\mathcal{R}$ and number $n \in \mathbb{N}$, the definition of $x_S \in \mathcal{D}^{|S|}$ does not change.

Basic coding theory and extractor definitions We let $\mathbb{F}_2$ denote the finite field of size two, and we let $\mathbb{F}_2^n$ denote a vector space over this field. The *Hamming weight* of a vector $x \in \mathbb{F}_2^n$ is defined as $\Delta(x) := \#\{i \in [n] : x_i = 1\}$, and the *Hamming distance* between two vectors $x, y \in \mathbb{F}_2^n$ is defined as $\Delta(x, y) := \Delta(x - y)$, where the subtraction is over $\mathbb{F}_2$. The *standard basis vectors* in $\mathbb{F}_2^n$ is the collection $\mathcal{E}^* := \{e_i\}_{i \in [n]}$, where $e_i \in \mathbb{F}_2^n$ holds a 1 at coordinate i and 0 everywhere else, and a *subcube* is a subspace spanned by some subset of $\mathcal{E}^*$. An (n, k, d)-code is a subset $Q \subseteq \mathbb{F}_2^n$ of size 2^k with the guarantee that any two distinct points $x, y \in Q$ have Hamming distance $\Delta(x, y) \geq d$. A linear $[n, k, d]$-code is simply an (n, k, d) code that is a subspace. Finally, we say that a source $\mathbf{X}$ over $\{0,1\}^n$ is an (n, k) source if it has min-entropy at least k, and we say that an extractor Ext an N-source extractor for entropy k if it is an extractor for a family of sources $\mathcal{X}$, where each $\mathbf{X} \in \mathcal{X}$ consists of N independent (n, k) sources.

Discrete probability In general, for a random variable $\mathbf{X} : \Omega \to V$, we are only concerned with the distribution over V induced by $\mathbf{X}$. We will therefore typically not define the outcome space Ω, and can assume it has any form we like (so long as the distribution induced by $\mathbf{X}$ does not change). Given random variables $\mathbf{X}, \mathbf{Y}$ and any $y \in \text{support}(\mathbf{Y})$, we let $(\mathbf{X} \mid \mathbf{Y} = y)$ denote a random variable that takes value x with probability $\Pr[\mathbf{X} = x \mid \mathbf{Y} = y]$. Given a random variable $\mathbf{X}$ and a family of random variables $\mathcal{Y}$, we say that $\mathbf{X}$ is a *convex combination* of random variables from $\mathcal{Y}$ if there exists a random variable $\mathbf{Z}$ such that for each $z \in \text{support}(\mathbf{Z})$, it holds that $(\mathbf{X} \mid \mathbf{Z} = z) \in \mathcal{Y}$. We define the *statistical distance* between two random variables $\mathbf{X}, \mathbf{Y}$ over V as

$$|\mathbf{X} - \mathbf{Y}| := \max_{S \subseteq V} |\Pr[\mathbf{X} \in V] - \Pr[\mathbf{Y} \in V]| = \frac{1}{2} \sum_{v \in V} |\Pr[\mathbf{X} = v] - \Pr[\mathbf{Y} = v]|,$$

and we say that $\mathbf{X}, \mathbf{Y}$ are ϵ*-close* if $|\mathbf{X} - \mathbf{Y}| \leq \epsilon$. Given these definitions, the following two standard facts are easy to show, and are extremely useful.

Fact 2.1. *For any random variable* $\mathbf{X} \sim \{0,1\}^m$ *and any constant* $c \in \{0,1\}^m$*, it holds that*

$$|\mathbf{X} - \mathbf{U}_m| = |(\mathbf{X} \oplus c) - \mathbf{U}_m|.$$

Fact 2.2. *For any random variables* $\mathbf{X}, \mathbf{Y}$*, where* $\mathbf{X} \sim \{0,1\}^m$*, it holds that*

$$|\mathbf{X} - \mathbf{U}_m| \leq \mathbb{E}_{y \sim \mathbf{Y}}[|(\mathbf{X} \mid \mathbf{Y} = y) - \mathbf{U}_m|].$$

Finally, we will need the following standard lemma about conditional min-entropy.

Lemma 2.3 ([MW97]). *Let* $\mathbf{X}, \mathbf{Y}$ *be random variables such that* $\mathbf{Y}$ *can take at most* ℓ *values. Then for any* $\epsilon > 0$*, it holds that*

$$\Pr_{y \sim \mathbf{Y}}[H_\infty(\mathbf{X} \mid \mathbf{Y} = y) \geq H_\infty(\mathbf{X}) - \log \ell - \log(1/\epsilon)] \geq 1 - \epsilon.$$

3 Explicit extremal designs via slicing codes and zero-sum sets

In this section, we efficiently construct designs with small independence number, and obtain our main theorem about explicit designs:

Theorem 3.1 (Theorem 1, restated)**.** *There exists an Algorithm $\mathcal{A}$ such that given any $n \geq r \geq s \in \mathbb{N}$ as input with r an even number, $\mathcal{A}$ runs in time* $\operatorname{poly}\left(\binom{n}{r}\right)$ *and outputs an (n, r, s)-design G with independence number*

$$\alpha(G) \leq C_{r,s} \cdot n^{\frac{2(r-s)}{r}}, \tag{1}$$

where $C_{r,s} = C \cdot r^4$ for some universal constant $C \geq 1$.

For all constants $r \geq s \in \mathbb{N}$ with r even, Theorem 3.1 constructs an explicit family of (n, r, s)-designs $(G_n)_{n \in \mathbb{N}}$ with small independence number. Like the non-explicit designs of Theorem 1.1 from [RŠ94], our derandomization focuses on the case where r, s are constant. However, it turns out that even for most *super-constant* r, s, our algorithm is still efficient. In particular, before proving Theorem 3.1, we make (and quickly prove) the following remark.

Remark 3.2. *Let $\mathcal{A}$ be the algorithm from Theorem 3.1, and let $m = m(n, r, s)$ be the number of hyperedges in the design produced by $\mathcal{A}$ on input (n, r, s). Then for any functions $r = r(n), s = s(n)$, Algorithm $\mathcal{A}$ is guaranteed to run in time* $\operatorname{poly}(n, m)$ *over the collection $\mathcal{I} = \{(n, r(n), s(n))\}_{n \in \mathbb{N}}$ as long as at least one of the following holds:*

- *The functions r, s are constant: $r(n) = O(1)$ and $s(n) = O(1)$; or*
- *There is a constant $\epsilon > 0$ such that inequality (1) is bounded above by $O(n^{1-\epsilon}), \forall (n, r, s) \in \mathcal{I}$.*

The first bullet in Remark 3.2 reiterates the fact that the algorithm in Theorem 3.1 is efficient when r, s are constant. The second bullet gives a more general remark on the performance of Algorithm $\mathcal{A}$ on super-constant r, s: it says that as long as Theorem 3.1 gave a "non-trivial" bound on the independence number in the first place, then the algorithm will run efficiently. This effectively covers all "interesting" regimes of r, s: indeed, the main application of selecting non-constant r, s would be to achieve independence bounds that are stronger than those achieved by constant r, s (and any constant r, s that achieve $\alpha(G) < n$ in Theorem 3.1 in fact achieve the second bullet).

To prove that Algorithm $\mathcal{A}$ is efficient given the condition in the second bullet, we use standard bounds on Turán numbers. The Turán number $T(n, \beta, r)$ is defined as the fewest number of edges in an r-uniform hypergraph with no independent set of size β, and it is known [Sid95] that $T(n, \beta, r) \geq \binom{n}{r}/\binom{\beta}{r}$. Thus, the second bullet implies the number of edges, $m = m(n, r, s)$, in the design is at least

$$T(n, Cn^{1-\epsilon} + 1, r) \geq T(n, n^{1-\epsilon/2}, r) \geq \binom{n}{r} \Big/ \binom{n^{1-\epsilon/2}}{r} \geq \binom{n}{r} \Big/ \binom{n}{r}^{1-\epsilon/4} \geq \binom{n}{r}^{\epsilon/4},$$

where we use the observation that the Turán number is non-increasing in its second argument, the fact that we can assume n, r are sufficiently large (since otherwise the efficiency claim is trivial), and a simple application of Stirling's formula. Thus, Algorithm $\mathcal{A}$ runs in time $\binom{n}{r} = \operatorname{poly}(n, m)$. In fact, since we gave a lower bound on m based on the independence number, it trivially holds that *any* algorithm that achieves independence numbers as small as $\mathcal{A}$ must output m edges, meaning that the runtime of $\mathcal{A}$ is optimal up to constant powers. This completes our discussion on Remark 3.2.

We now turn to proving Theorem 3.1. We start with the simple observation that hypergraphs over n vertices can be identified with subsets of $\mathbb{F}_2^n$. In particular, any subset $T \subseteq \mathbb{F}_2^n$ induces a hypergraph $G_T = (V, E)$ in the following way: identify V with $[n]$, and for each $x \in T$ add a hyperedge $e \subseteq [n]$ to E that contains exactly the coordinates that take the value 1 in x. Using this correspondence, we can instead focus on constructing special subsets of $\mathbb{F}_2^n$, and thereby leverage the tools of linear algebra and coding theory.

To obtain our designs, we will need to explicitly construct a subset $T \subseteq \mathbb{F}_2^n$ such that (1) G_T is an (n, r, s)-design; and (2) G_T has small independence number. We can make sure this happens via the following two simple facts, which describe how these hypergraph properties can be identified with properties of subsets in $\mathbb{F}_2^n$.

Fact 3.3. *For any subset $T \subseteq \mathbb{F}_2^n$, the hypergraph G_T is an (n, r, s)-design if and only if (i) every $x \in T$ has $\Delta(x) = r$; and (ii) any two distinct $x, y \in T$ have $\Delta(x, y) > 2(r - s)$.*

Proof. The two conditions are sufficient because the first one guarantees that G_T will be r-uniform, and the second one guarantees that any two edges in G_T intersect at $< s$ points. They are both necessary because if the first does not hold, G_T will not be r-uniform, and if the first holds but the second does not, then two edges will end up sharing $\geq s$ points. □

Fact 3.4. *For any subset $T \subseteq \mathbb{F}_2^n$, the hypergraph G_T has independence number $\alpha(G_T) < \ell$ if and only if every subcube $A \subseteq \mathbb{F}_2^n$ of dimension at least ℓ has at least one point in T.*

Proof. If $\alpha(G_T) \geq \ell$, there is an independent set $S \subseteq V = [n]$ of size at least ℓ, and thus the subcube $A := \mathit{span}(\{e_i\}_{i \in S})$ of dimension ℓ has no points in T. If there is a subcube $A \subseteq \mathbb{F}_2^n$ of dimension ℓ with no points in T, the set $S \subseteq [n]$ indexing the standard basis vectors that span A must have size ℓ and constitute an independent set in G_T. □

By Fact 3.3 and Fact 3.4, we see that the task of constructing an (n, r, s)-design G with small independence number is *equivalent* to the task of constructing a subset $T \subseteq \mathbb{F}_2^n$ with the following *three properties*:

1. T lies in the Hamming slice $\Delta_r := \{x \in \mathbb{F}_2^n : \Delta(x) = r\}$,

2. Points in T have pairwise Hamming distance $> 2(r - s)$, and

3. Any subcube of *relatively small* dimension intersects T.

In order to construct a set $T \subseteq \mathbb{F}_2^n$ with these three properties, we use connections to *coding theory* and *zero-sum problems*. In particular, recall that an (n, k, d)-code is a subset $Q \subseteq \mathbb{F}_2^n$ of size 2^k with the guarantee that any two distinct points $x, y \in Q$ have Hamming distance $\Delta(x, y) \geq d$. Thus, if we take any (n, k, d)-code $Q \subseteq \mathbb{F}_2^n$ with $d > 2(r - s)$ and intersect it with the Hamming slice Δ_r, we obtain a set $T = Q \cap \Delta_r$ that enjoys properties (1) and (2). In order to endow it with property (3), we will need to start with some code Q such that for any relatively large subcube S, the set $S \cap T = S \cap (Q \cap \Delta_r) = (S \cap Q) \cap \Delta_r$ is non-empty.

The trick here is to start with a *linear* code Q. A *linear* $[n, k, d]$-code $Q \subseteq \mathbb{F}_2^n$ is simply an (n, k, d) code that is also a subspace. The condition $(S \cap Q) \cap \Delta_r \neq \emptyset$ required for property (3) now becomes more concrete: since Q is a subspace, $S \cap Q$ is also a subspace, and thus we can make sure it contains some vector of Hamming weight r as long as we can show that *every* large subspace contains such a vector. In particular, defining $\Lambda_r(n)$ to be the dimension of the largest subspace $R \subseteq \mathbb{F}_2^n$ containing no vector of Hamming weight r, we prove the following lemma.

Lemma 3.5. *If $Q \subseteq \mathbb{F}_2^n$ is a linear $[n,k,d]$-code with $d > 2(r-s)$, then the hypergraph $G_{Q\cap\Delta_r}$ is an (n,r,s)-design with independence number $\alpha = \alpha(G_{Q\cap\Delta_r})$ that obeys the following inequality:*

$$\alpha - \Lambda_r(\alpha) \leq n - k$$

Proof. It follows immediately from Fact 3.3 that $G_{Q\cap\Delta_r}$ is an (n,r,s)-design. By Fact 3.4, there is a subcube $A = span(e_{i_1}, \dots, e_{i_\alpha}) \subseteq \mathbb{F}_2^n$ of dimension α that doesn't intersect $Q \cap \Delta_r$. Thus, if we define $A' := A \cap Q$, then A' contains no vector of Hamming weight r, and furthermore it has dimension $dim(A') = dim(A \cap Q) \geq dim(A) + dim(Q) - n = \alpha + k - n$. Notice now that if we define the projection $\pi : \mathbb{F}_2^n \to \mathbb{F}_2^\alpha$ as the map $(x_1, \dots, x_n) \mapsto (x_{i_1}, \dots, x_{i_\alpha})$, then the subset $\pi(A')$ is still a subspace (albeit now of $\mathbb{F}_2^\alpha$) of dimension $dim(\pi(A')) \geq \alpha + k - n$ containing no vector of Hamming weight r. Thus, by definition of Λ_r, it must hold that $\alpha + k - n \leq dim(\pi(A')) \leq \Lambda_r(\alpha)$. □

In order to construct an (n,r,s)-design from Lemma 3.5 with the smallest possible independence number α, we will want an explicit $[n,k,d > 2(r-s)]$-linear code with the largest possible dimension k, along with a strong upper bound on $\Lambda_r(n)$. We start with the latter.

Getting a good upper bound on $\Lambda_r(n)$ is closely related to the theory of *zero-sum problems*. In this field, one parameter of great interest is the (generalized) *Erdős-Ginzburg-Ziv constant*(s) of a finite abelian group. Given $n \geq r \in \mathbb{N}$ where r is even, this parameter is defined for $\mathbb{F}_2^n$ as the smallest integer $s_r(n)$ such that any *sequence* of $s_r(n)$ values in $\mathbb{F}_2^n$ contains a subsequence of length r that sums to zero. For our application, it will be more convenient to use an almost identical parameter $\beta_r(n)$, defined as the size of the largest *subset* of $\mathbb{F}_2^n$ containing no r elements that sum to zero. Using slightly different terminology, the relationship between $\beta_r(n)$ and $\Lambda_r(n)$ was shown in [Sid20]. We include it here, in our language, for completeness.

Lemma 3.6 ([Sid20])**.** *For every $n \geq r \in \mathbb{N}$ where r is even,*

$$\beta_r(n - \Lambda_r(n)) \geq n.$$

Proof. Let $R \subseteq \mathbb{F}_2^n$ be a subspace of dimension $k := \Lambda_r(n)$, and define $d := n - k$. Let $v_1, \dots, v_d$ be a basis for the orthogonal complement of R, and define the matrix $M \in \mathbb{F}_2^{d\times n}$ so that its i^{th} row is v_i. Notice that R contains exactly the solutions to $Mx = 0$, and thus R has a vector of Hamming weight r if and only if there are r columns in M that sum to zero. By definition of $\Lambda_r(n)$, we know R has no such vector, and thus $n \leq \beta_r(d) = \beta_r(n - \Lambda_r(n))$. □

To get a good upper bound on $\Lambda_r(n)$, we need a good upper bound on $\beta_r(n)$. In 2018, Sidorenko provided a very strong bound of this type:

Theorem 3.7 ([Sid18], Theorem 4.4)**.** *There is a universal constant $C > 0$ such that for every $n, r \in \mathbb{N}$ where r is even,*

$$\beta_r(n) \leq C \cdot r^3 \cdot 2^{2n/r}.$$

By plugging this bound into Lemma 3.6, we get the following corollary.

Corollary 3.8 ([Sid20])**.** *There is a universal constant $C > 0$ such that for any $n \geq r \in \mathbb{N}$ where r is even, the largest subspace $S \subseteq \mathbb{F}_2^n$ with no vector of Hamming weight r has dimension*

$$\Lambda_r(n) \leq n - (r \log n - 3r \log r - r \log C)/2.$$

We are finally ready to prove our main design lemma, which reduces the problem of constructing (n,r,s)-designs with small independence number to constructing high-dimensional linear codes.

Lemma 3.9 (Main design lemma)**.** *There is a universal constant $C > 0$ such that for every $n \geq r \geq s$ with r even, if $Q \subseteq \mathbb{F}_2^n$ is a linear $[n, k, d]$-code with $d > 2(r - s)$, then $G_{Q \cap \Delta_r}$ is an (n, r, s)-design with independence number*

$$\alpha(G_{Q\cap\Delta_r}) \leq C \cdot r^3 \cdot 2^{2(n-k)/r}.$$

Proof. Simply plug the bound on $\Lambda_r(\alpha)$ from Corollary 3.8 into Lemma 3.5. □

To complete the proof of Theorem 3.1, we now just need to explicitly construct a linear code with very high dimension. In 1959-1960, Bose, Ray-Chaudhuri [BRC60], and Hocquenghem [Hoc59] explicitly constructed codes of exactly this type (see [GB10] for a great exposition of these codes, which are known as *BCH codes*). In particular, they proved the following theorem.

Theorem 3.10 ([BRC60, Hoc59])**.** *For every $m, t \in \mathbb{N}$, there exists an $[n, k, d]$-linear code $\mathbf{BCH}_{m,t} \subseteq \mathbb{F}_2^n$ with block length $n = 2^m - 1$, dimension $k \geq n - mt$, and distance $d > 2t$. Furthermore, there exists an Algorithm $\mathcal{B}$ that given any $m, t \in \mathbb{N}$ and $x \in \mathbb{F}_2^n$ as input, checks if $x \in \mathbf{BCH}_{m,t}$ in* $\mathrm{poly}(n)$ *time.*

By instantiating Lemma 3.9 with Theorem 3.10, we can finally prove Theorem 3.1.

Proof of Theorem 3.1. We start by assuming that $n = 2^m - 1$ for some $m \in \mathbb{N}$. Then, we let $t = r - s$, and use Theorem 3.10 to define the $[n, k, d]$-linear code $Q := \mathbf{BCH}_{m,t} \subseteq \mathbb{F}_2^n$, where $k \geq n - mt = n - m(r - s)$ and $d > 2t = 2(r - s)$. By Lemma 3.9, we know that $G_{Q\cap\Delta_r}$ is an (n, r, s)-design with independence number

$$\alpha(G_{Q\cap\Delta_r}) \leq C \cdot r^3 \cdot 2^{2(n-k)/r} \leq C \cdot r^3 \cdot 2^{2mt/r} = C \cdot r^3 \cdot (2^m)^{2(r-s)/r} \leq 2C \cdot r^3 \cdot n^{2(r-s)/r}.$$

Note that $G_{Q\cap\Delta_r}$ can be constructed in $\mathrm{poly}(\binom{n}{r})$ time if $Q \cap \Delta_r$ can be constructed in $\mathrm{poly}(\binom{n}{r})$ time, and this can be done by simply checking (and appropriately including) whether each of the $\binom{n}{r}$ elements in Δ_r belong to Q, using Algorithm $\mathcal{B}$ from Theorem 3.10.

If n is of the form 2^m, we can follow the previous procedure to draw hyperedges around the first $n - 1$ vertices, and then add one more isolated vertex (contained in no edges) at the end to finish the hypergraph. Clearly we will still have $\alpha(G_{Q\cap\Delta_r}) \leq 3C \cdot r^3 \cdot n^{2(r-s)/r}$.

If n is not of the form $2^m - 1$ nor 2^m, then it can be written as a sum $x_0 2^0 + \cdots + x_d 2^d$ over $\mathbb{N}$, where $d = \lceil \log n \rceil$ and each $x_i \in \{0, 1\}$. We can then follow the most recent procedure to construct a graph G_i over 2^i vertices separately for each nonzero x_i. The final graph $G = \bigcup_i G_i$ is clearly still an (n, r, s)-design, and it has independence number

$$\begin{aligned}\alpha(G) = \sum_{i:x_i=1} \alpha(G_i) \leq \sum_{0\leq i\leq\lceil\log n\rceil} 3C \cdot r^3 \cdot (2^i)^{2(r-s)/r} &= 3C \cdot r^3 \sum_{0\leq i\leq\lceil\log n\rceil} (2^i)^{2(r-s)/r} \\ &= 3C \cdot r^3 \cdot \frac{(2^{\lceil\log n\rceil+1})^{2(r-s)/r} - 1}{2^{2(r-s)/r} - 1}.\end{aligned}$$

It is straightforward to verify that for a large enough universal constant C', the above fraction is bounded above by $C' \cdot r \cdot n^{2(r-s)/r}$, which completes the proof. □

4 Extractors for adversarial sources via designs and LREs

In this section, we will show how to combine our designs from Section 3 with a very recent object, known as a *leakage-resilient extractor* (LRE), in order to obtain improved extractors for 0-local adversarial sources (as defined by Definition 1.3). We will prove our second main theorem:

Theorem 4.1 (Theorem 2, restated)**.** *There is a universal constant $C > 0$ such that for any fixed $\delta > 0$ and all sufficiently large $N, K, n, k \in \mathbb{N}$ satisfying $k \geq \log^C n$ and $K \geq N^\delta$, there exists an explicit extractor $\mathsf{Ext} : (\{0,1\}^n)^N \to \{0,1\}^m$ for (N, K, n, k)-adversarial sources of locality 0, with output length $m = k^{\Omega(1)}$ and error $\epsilon = 2^{-k^{\Omega(1)}}$.*

Previously, the best explicit extractor for this setting was constructed by Chattopadhyay et al. [CGGL20], and required $K \geq N^{0.5+o(1)}$ good sources. On the other hand, it is easy to give a *non-explicit* extractor that requires just $K \geq 2$ good sources.[4] Thus, while our explicit constructions greatly improve the state-of-art (and most notably break the "$\sqrt{N}$ barrier"), there is still a lot of room for improvement. Further improvement, however, will require significantly new techniques.

Our construction leverages the "*activation vs. fragile correlation*" paradigm introduced in [CGGL20] for extracting from adversarial sources. This paradigm was first introduced in an attempt to construct a low-error extractor for (N, K, n, k)-adversarial sources of locality 0, given just $k \geq \operatorname{polylog} n$ entropy and as a few good sources, K, as possible. Since there exists a three-source extractor Ext_0 for $k_0 \geq \operatorname{polylog} n$ entropy and exponentially small error [Li15], a natural idea is to somehow employ this object as a subroutine. Using this idea, [CGGL20] proposed an extractor for adversarial sources that works as follows. Given as input an adversarial source $\mathbf{X} = (\mathbf{X}_1, \ldots, \mathbf{X}_N)$, the extractor carefully selecting triples of sources, calls Ext_0 over each triple, and XORs the results. [CGGL20] argued that this procedure outputs uniform bits as long as the following two properties hold:

1. **Activation**: some Ext_0 call is *activated*, i.e., only given good sources as input.
2. **Fragile correlation**: fixing the (XOR of the) output of all other Ext_0 calls does not affect the output of the activated Ext_0 call (with high probability).

It is not hard to see why these conditions suffice: *activation* guarantees that some Ext_0 call outputs uniform bits, while *fragile correlation* guarantees that these uniform bits will be propagated through to the overall output of the extractor (by Fact 2.1 and Fact 2.2). Thus, the main challenge considered in [CGGL20] is determining *how to select triples* such that activation and fragile correlation are guaranteed.

The key idea in [CGGL20] is to select triples using the hyperedges of a 3-uniform hypergraph, $G = (V, E)$. Then, we know that activation is guaranteed as long as the good sources *cover* some hyperedge $e \in E$, which is guaranteed to happen whenever $K > \alpha(G)$. In order to ensure fragile correlation, [CGGL20] observed that it suffices to require that $G = (V, E)$ is a *partial Steiner triple system*, also known as an $(N, 3, 2)$-design. Such a hypergraph guarantees that each Ext_0 call shares at most one source with the activated Ext_0 call. Thus, if we start by fixing all sources that are *not* inputs to the activated Ext_0 call, it is then easy to fix the outputs of all other Ext_0 calls without introducing correlation between the inputs to the activated call. Furthermore, by Lemma 2.3, we can show that this process barely decreases the entropy of the inputs to the activated call, and thus its output remains uniform.

We have shown that the construction above provides a low-error extractor for (N, K, n, k)-adversarial sources of locality 0, where $k \geq \operatorname{polylog} n$ and $K > \alpha(G)$. Thus, the goal becomes to explicitly construct an $(N, 3, 2)$-design $G = (V, E)$ with small independence number. Chattopadhyay et al. [CGGL20] construct such an object with $\alpha(G) < N^{0.923}$, and thus gave an explicit extractor when there are $K \geq N^{0.923}$ good sources. In order to improve this requirement on K, it

[4] This extractor calls an optimal two-source extractor over every pair of sources in the adversarial source, and takes the XOR of the results [CL16]. To see why this works, we refer the reader to a similar proof sketch for a slightly more involved construction, provided in the following paragraphs.

is natural to try to construct an $(N, 3, 2)$-design with smaller independence number. However, this seems difficult, and furthermore the tightness of Theorem 1.1 implies that this technique cannot possibly give an extractor that requires fewer than $K \geq N^{0.5+o(1)}$ good sources.

Chattopadhyay et al. [CGGL20] take a different approach. They observed that the above extractor, based on $(N, 3, 2)$-designs, worked for the following primary reason: if a three-source extractor outputs uniform bits on input $\mathbf{X}_1, \mathbf{X}_2, \mathbf{X}_3$, then the output still looks uniform *even conditioned on* the output of several functions, each acting on *just one of* $\mathbf{X}_1, \mathbf{X}_2, \mathbf{X}_3$. They suggested that if one could construct a three-source extractor with stronger conditioning properties, then perhaps they could start with a 3-uniform hypergraph from a more general class $\mathcal{H}$ than $(N, 3, 2)$-designs. The advantage behind such an approach is that it could be easier to explicitly construct a hypergraph from $\mathcal{H}$ with small independence number, and thereby improve the requirement on K.

By applying a *strong two-source condenser* from [BACDTS19], Chattopadhyay et al. [CGGL20] explicitly construct more robust three-source (and multi-source) extractors with stronger conditioning properties. By combining these objects with various types of explicit hypergraphs, they successfully reduce the requirement on good sources from $K \geq N^{0.923}$ to $K \geq N^{0.5+o(1)}$. Unfortunately, however, the conditioning properties of their robust subroutine extractors are extremely specific, and as a result they can only be combined with very specialized types of hypergraphs. These hypergraphs offer no clean generalization of $(N, 3, 2)$-designs, and furthermore they appear to be too specialized to offer any further improvement on K (and, in particular, break the "$\sqrt{N}$ barrier").

If one hopes to significantly improve K, it appears that one would need a multi-source extractor with *even stronger* conditioning properties to use as a subroutine. Recently, exactly such an object was defined in [CGG+20], and is known as a *leakage-resilient extractor* (LRE). LREs are very general objects with extremely strong conditioning properties. The exact variant that will be useful here is actually a specialization known as *extractors for cylinder intersections*, first introduced in [KMS19]. Informally, we define an (r, s)*-leakage-resilient extractor* to be an r-source extractor LRE that outputs bits that look uniform, *even conditioned on* the output of several functions that each act on *fewer than* s of the inputs to LRE. Formally, it is defined as follows.

Definition 4.2 ([KMS19, CGG+20])**.** *A function* $\mathsf{LRE} : (\{0,1\}^n)^r \to \{0,1\}^m$ *is an* (r, s)-leakage-resilient extractor *for entropy* k *and error* ϵ *if the following holds. Let* $\mathbf{X} := (\mathbf{X}_1, \ldots, \mathbf{X}_r)$ *be any* r *independent* (n, k) *sources, let* $\mathcal{T} := \binom{[N]}{s-1}$*, and let* $\mathcal{L} := \{\mathsf{Leak}_T : (\{0,1\}^n)^{s-1} \to \{0,1\}^m\}_{T \in \mathcal{T}}$ *be any collection of functions. Then:*

$$|\mathsf{LRE}(\mathbf{X}) \circ (\mathsf{Leak}_S(\mathbf{X}_S))_{S \in \mathcal{S}} - \mathbf{U}_m \circ (\mathsf{Leak}_S(\mathbf{X}_S))_{S \in \mathcal{S}}| \leq \epsilon.$$

Given such a robust extractor, it is now easy to generalize the original extractor of [CGGL20] in a clean, natural way: instead of calling a three-source extractor over the hyperedges of an $(N, 3, 2)$-design and XORing the results, we call an (r, s)-leakage-resilient extractor over the hyperedges of an (N, r, s)-design and XOR the results. Once again, we can ensure *activation* as long as the number of good sources, K, exceeds the independence number of the design. On the other hand, instead of using Lemma 2.3 to ensure fragile correlation, we simply use the leakage-resilience of our leakage-resilient extractor: to see why this works, simply observe that an (N, r, s)-design guarantees that the intersection of two hyperedges has size $< s$, while a leakage-resilient extractor guarantees to output uniform bits *even conditioned on* several leaks that each act on $< s$ of its inputs.

Formally, we prove the following lemma, which provides a framework for combining leakage-resilient extractors with general designs in order to extract from adversarial sources.

Lemma 4.3. *Let $G = ([N], E)$ be an (N, r, s)-design with independence number α, and let $\mathsf{Ext}_0 : (\{0,1\}^n)^r \to \{0,1\}^m$ be an (r, s)-leakage resilient extractor for entropy k_0 with error ϵ_0. Then for any $K > \alpha$ and $k \geq k_0$, the function $\mathsf{Ext}_G : (\{0,1\}^n)^N \to \{0,1\}^m$ defined as*

$$\mathsf{Ext}_G(X) := \bigoplus_{e \in E(G)} \mathsf{Ext}_0(X_e)$$

is an extractor for (N, K, n, k) adversarial sources of locality 0 with error $\epsilon = \epsilon_0$.

Proof. Let $\mathbf{X}$ be an (N, K, n, k) adversarial source. We must show that $|\mathsf{Ext}_G(\mathbf{X}) - \mathbf{U}_m| \leq \epsilon$. Because $K > \alpha$, there is some $e^* \in E$ containing only good sources, i.e., $\mathbf{X}_i$ has entropy at least k for each $i \in e^*$. Without loss of generality, we assume $e^* = [r]$. We now fix all other sources $\mathbf{Z}_1 = (\mathbf{X}_j)_{j \notin e^*}$, using Fact 2.2:

$$|\mathsf{Ext}_G(\mathbf{X}) - \mathbf{U}_m| \leq \mathbb{E}_{z_1 \sim \mathbf{Z}_1}[|(\mathsf{Ext}_G(\mathbf{X}) \mid \mathbf{Z}_1 = z_1) - \mathbf{U}_m|].$$

Consider any $z_1 = (x_j)_{j \notin e^*}$. For each $e \in E(G)$, we define the restriction $\mathsf{Ext}_0^e : (\{0,1\}^n)^{|e \cap e^*|} \to \{0,1\}^m$ as $\mathsf{Ext}_0^e(Y_1, \ldots, Y_{|e \cap e^*|}) = \mathsf{Ext}_0(Y_1, \ldots, Y_{|e \cap e^*|}, (x_j)_{j \in e \setminus e^*})$, so that we may write

$$(\mathsf{Ext}_G(\mathbf{X}) \mid \mathbf{Z} = z_1) = \bigoplus_{e \in E(G)} \mathsf{Ext}_0^e(\mathbf{X}_{e \cap e^*}) = \mathsf{Ext}_0(\mathbf{X}_{e^*}) \oplus \bigoplus_{e \in E(G) \setminus \{e^*\}} \mathsf{Ext}_0^e(\mathbf{X}_{e \cap e^*}).$$

Because G is an (N, r, s)-design, any two edges share at most $s - 1$ vertices. Thus, we may partition $E(G) \setminus \{e^*\}$ into $\binom{r}{s-1}$ sets, depending on the intersection behavior of each edge with e^*. In particular, for each $S \in \binom{e^*}{s-1}$, we define:

$$\mathcal{W}_S := \{e \in E : e \cap e^* \subseteq S\}.$$

If any $e \in E$ ends up in more than one $\mathcal{W}_S$, we simply remove it from all but one of these sets. Now, for each $S \in \binom{e^*}{s-1}$, we define $\mathsf{Leak}_S : (\{0,1\}^n)^{s-1} \to \{0,1\}^m$ such that for any $X \in (\{0,1\}^n)^N$, $\mathsf{Leak}_S(X_S) = \bigoplus_{e \in \mathcal{W}_S} \mathsf{Ext}_0^e(X_{e \cap e^*})$, which is a valid definition because $e \cap e^*$ is always in S, by definition of $\mathcal{W}_S$. We may now write

$$(\mathsf{Ext}_G(\mathbf{X}) \mid \mathbf{Z}_1 = z_1) = \mathsf{Ext}_0(\mathbf{X}_{e^*}) \oplus \bigoplus_{S \in \binom{e^*}{s-1}} \mathsf{Leak}_S(\mathbf{X}_S). \tag{2}$$

To bound the distance of this random variable from uniform, we now define the second random variable we will fix, $\mathbf{Z}_2 := (\mathsf{Leak}_S(\mathbf{X}_S))_{S \in \binom{e^*}{s-1}}$. Fixing this random variable, we have:

$$\begin{aligned} |(\mathsf{Ext}_G(\mathbf{X}) \mid \mathbf{Z}_1 = z_1) - \mathbf{U}_m| &\leq \mathbb{E}_{z_2 \sim \mathbf{Z}_2}[|(\mathsf{Ext}_G(\mathbf{X}) \mid \mathbf{Z}_1 = z_1, \mathbf{Z}_2 = z_2) - \mathbf{U}_m|] \\ &= \mathbb{E}_{z_2 \sim \mathbf{Z}_2}[|(\mathsf{Ext}_0(\mathbf{X}_{e^*}) \mid \mathbf{Z}_2 = z_2) - \mathbf{U}_m|] \\ &= |\mathsf{Ext}_0(\mathbf{X}_{e^*}) \circ \mathbf{Z}_2 - \mathbf{U}_m \circ \mathbf{Z}_2|, \end{aligned}$$

where the first and last (in)equalities follow easily from the definition of statistical distance, and the second (in)equality follows from Equation (2) and the fact that adding a constant to a random variable doesn't change its distance from uniform. But notice that by definition of $\mathbf{Z}_2$ and the leakage-resilience of Ext_0, this quantity is bounded above by ϵ_0, which completes the proof. □

In order to highlight the generality of this framework, we observe that by Lemma 2.3, a standard three-source extractor is, in fact, a $(3, 2)$-leakage-resilient extractor (up to some negligible loss in parameters). Thus, by instantiating Lemma 4.3 with $r = 3, s = 2$, we recover the original extractor and analysis of [CGGL20]. Even better, since Theorem 1.1 tells us that the independence number α of an (N, r, s)-design decreases quickly as r, s grow large together, we see that Lemma 4.3 offers a concrete way to construct extractors for adversarial sources with much fewer good sources, K.

If we want to realize the above plan, we need two explicit objects. First, we need explicit (N, r, s)-designs with independence numbers that decrease quickly as r, s grow together. Theorem 1 of the current paper gives exactly this, and in fact the independence numbers of our designs decrease with r, s *almost as quickly as possible*, as shown by the tightness of Theorem 1.1.

Second, we need explicit leakage-resilient extractors for polylogarithmic entropy that have exponentially small error. Very recently, these exact objects were constructed:

Theorem 4.4 ([CGG$^+$20])**.** *There is a universal constant $C > 0$ such that for any sufficiently large* constant $r \in \mathbb{N}$ *and all $n, k \in \mathbb{N}$ satisfying $k \geq \log^C n$, there exists an explicit $(r, r-1)$-leakage resilient extractor* $\mathsf{Ext} : (\{0,1\}^n)^r \to \{0,1\}^m$ *for min-entropy k with output length $m = k^{\Omega(1)}$ and error $\epsilon = 2^{-k^{\Omega(1)}}$.*

By combining these explicit LREs with our explicit designs, we can finally prove Theorem 4.1, which significantly improves the adversarial source extractors of [CGGL20].

Proof of Theorem 4.1. Let C be the same universal constant from Theorem 4.4, and let $r \in \mathbb{N}$ be a sufficiently large (even) constant such that $2/r < \delta$, and such that Theorem 4.4 guarantees the existence of an explicit $(r, r-1)$-leakage resilient extractor $\mathsf{Ext}_0 : (\{0,1\}^n)^r \to \{0,1\}^m$ for min-entropy $k \geq \log^C n$ with output length $m = k^{\Omega(1)}$ and error $\epsilon = 2^{-k^{\Omega(1)}}$. For sufficiently large $N \in \mathbb{N}$, Theorem 1 guarantees the existence of an $(N, r, r-1)$-design G with independence number $\alpha < N^\delta$ that is computable in $\text{poly}(\binom{N}{r}) = \text{poly}(N)$ time. The result now follows by instantiating Lemma 4.3 with Ext_0 and G. □

5 Extractors for small-space sources via adversarial sources

In this section, we will show how to use our extractors from Section 4 to obtain better extractors for small-space sources (as defined by Definition 1.4). We will prove our third main theorem:

Theorem 5.1 (Theorem 3, restated)**.** *For any fixed $\delta \in (0, 1/2]$ there is a constant $C > 0$ such that for all $n, k, s \in \mathbb{N}$ satisfying $k \geq Cn^{1/2+\delta}s^{1/2-\delta}$, there exists an explicit extractor* $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^m$ *for space s sources of min-entropy k, with output length $m = n^{\Omega(1)}$ and error $\epsilon = 2^{-n^{\Omega(1)}}$.*

Until very recently, the best explicit extractor for this setting [KRVZ06] required entropy $k \geq Cn^{1-\gamma}s^\gamma$, where $\gamma > 0$ is a tiny constant and C is a large one. In [CGGL20], this requirement was significantly improved to $k \geq Cn^{2/3+\delta}s^{1/3-\delta}$, for an arbitrarily small constant $\delta > 0$, and the current paper (Theorem 5.1) further improves this to $k \geq Cn^{1/2+\delta}s^{1/2-\delta}$. We note that this line of improvements is *strict* in terms of both entropy *and* space, via the following remark.

Remark 5.2. *For all $s = s(n)$, the requirement $k \geq Cn^{1-\epsilon}s^\epsilon$* strictly decreases *as ϵ grows. This is because the requirement is trivial whenever $s \geq n$; we may therefore assume $s < n$, and so shifting the weight of the cumulative power from n to s strictly improves the requirement.*

Non-constructively, it is known [KRVZ06] that there exist extractors for space s sources of min-entropy $k \geq O(s + \log n)$ that have error $\epsilon = 2^{-\Omega(k)}$. Furthermore, in the large-error setting, Chattopadhyay and Li [CL16] constructed an extractor for small-space sources that requires just $k \geq n^{o(1)}$ entropy, but has error $\epsilon = n^{-\Omega(1)}$. Thus, while Theorem 5.1 significantly improves the state-of-art in low-error extraction, there is still a lot of room for improvement. However, we note (in Remark 5.7) that any substantial improvements to our low-error extractors will require significantly new techniques.

We now proceed to prove Theorem 5.1. The techniques that follow, which will reduce the task of extracting from small space sources to the task of extracting from adversarial sources, are just slightly optimized versions of the exact arguments that appear in [CGGL20]. However, we include them here for completeness.

The first step is to reduce small-space sources to a class of sources known as *total entropy sources*, defined as follows.

Definition 5.3. *A random variable* $\mathbf{X}$ *over* $(\{0,1\}^\ell)^r$ *is an* (r, ℓ, k)-total entropy source *if* $\mathbf{X} = (\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_r)$*, where each* $\mathbf{X}_i$ *is an independent source over* $\{0,1\}^\ell$*, and* $\sum_{i\in[r]} H_\infty(\mathbf{X}_i) \geq k$.

In [KRVZ06], Kamp et al. showed that upon fixing a few positions in the random walk that generates a small space source $\mathbf{X}$, it is straightforward to use Lemma 2.3 to show that $\mathbf{X}$ becomes a total-entropy source, with high probability. We include the proof for completeness.

Lemma 5.4 ([KRVZ06]). *Let* $\mathbf{X}$ *be a space* s *source over* $\{0,1\}^n$ *with min-entropy* k*. Then for any* $\alpha \in (0, 1/4]$ *such that* $r = \alpha k/s$ *and* $\ell = ns/(\alpha k)$ *are positive integers, it holds that* $\mathbf{X}$ *is* $2^{-k/4}$*-close to a convex combination of* $(r, \ell, k/2)$*-total entropy sources.*

Proof. For each $i \in [n]$, let $\mathbf{W}_i \sim \{0,1\}^s$ be the random variable denoting the state reached in layer i of the branching program in the random walk that generates $\mathbf{X}$. Observe that fixing any $\mathbf{W}_i$ breaks $\mathbf{X}$ into two independent sources. More generally, observe that if we define $\mathbf{W}^* := (\mathbf{W}_{i\ell+1})_{i\in[0,r-1]}$, then if we condition $\mathbf{X}$ on any fixing of $\mathbf{W}^*$, it must hold that $\mathbf{X}$ becomes an (r, ℓ, Γ)-total entropy source, for *some* Γ. Furthermore, by Lemma 2.3, we know

$$\Pr_{w\sim\mathbf{W}^*}[H_\infty(\mathbf{X} \mid \mathbf{W}^* = w) \geq k - rs - k/4 = k - \alpha k - k/4 \geq k/2] \geq 1 - 2^{-k/4}. \tag{3}$$

Thus, the random variable $(\mathbf{X} \mid \mathbf{W}^* = w)$ is an $(r, \ell, k/2)$-total entropy source with probability at least $1 - 2^{-k/4}$ over $w \sim \mathbf{W}^*$, which completes the proof. □

The next step is to show that a total-entropy source looks like an adversarial source of locality 0, using a standard Markov-type argument:

Lemma 5.5. *Let* $\mathbf{X}$ *be an* (r, ℓ, Γ)*-total entropy source. Then for any* $N, K, n, k \in \mathbb{N}$ *with* $Nn = r\ell$ *and* n *a multiple of* ℓ*,* $\mathbf{X}$ *is also an* (N, K, n, k)*-adversarial source of locality 0, as long as* $Kn + Nk \leq \Gamma$.

Proof. By definition of total-entropy source, $\mathbf{X} = (\mathbf{X}_1, \ldots, \mathbf{X}_r)$, where each $\mathbf{X}_i$ is an independent source over $\{0,1\}^\ell$. By collecting the sources $\mathbf{X}_i$ into N buckets containing n/ℓ sources each, we see that $\mathbf{X}$ is also an (N, n, Γ)-total entropy source, and may be rewritten as $\mathbf{X} = (\mathbf{X}_1, \ldots, \mathbf{X}_N)$, where each $\mathbf{X}_i$ is an independent source over $\{0,1\}^n$. If $\mathbf{X}$ were not an (N, K, n, k)-adversarial source of locality 0, then the $K - 1$ highest entropy sources in $\mathbf{X}$ each have entropy at most n, and the remaining each have entropy $< k$. This yields $H_\infty(\mathbf{X}) = \Gamma < (K-1)n + (N-(K-1))k < Kn + Nk$, contradicting the given lower bound on Γ. □

Given the above reduction, we can now use our improved adversarial source extractors (from Theorem 2) to give improved extractors for total-entropy sources.

Theorem 5.6. *For any fixed $\delta > 0$ and all sufficiently large $r, \ell, \Gamma \in \mathbb{N}$ with $\Gamma \geq \max\left\{(r\ell)^{1/2+\delta}, r^{\delta}\ell\right\}$, there exists an explicit extractor $\mathsf{Ext} : (\{0,1\}^{\ell})^{r} \to \{0,1\}^{m}$ for (r, ℓ, Γ)-total entropy sources, with output length $m = (r\ell)^{\Omega(1)}$ and error $\epsilon = 2^{-(r\ell)^{\Omega(1)}}$.*

Proof. Fix any $N, n \in \mathbb{N}$ such that $Nn = r\ell$ and n is a multiple of ℓ. By Lemma 5.5, every (r, ℓ, Γ)-total entropy source is also an (N, K, n, k)-adversarial source of locality 0, provided $Kn + Nk \leq \Gamma$. Thus, by Theorem 2, for any fixed $\delta_0 > 0$ there exists an explicit extractor $\mathsf{Ext}_0 : (\{0,1\}^{r})^{\ell} \to \{0,1\}^{m}$ for (r, ℓ, Γ)-total entropy sources with output length $m = n^{\Omega(1)}$ and error $\epsilon = 2^{-n^{\Omega(1)}}$, provided $N^{\delta_0}n + Nn^{\delta_0} \leq \Gamma$ and N, n are sufficiently large. To achieve the parameters claimed in the theorem statement, pick $\delta_0 = \delta/2$ and set N, n as follows: (i) if $r \geq \ell$, set $N = n = \sqrt{r\ell}$; (ii) if $r < \ell$, set $N = r$ and $n = \ell$. We conclude by remarking that this casework was motivated by trying to minimize the requirement on Γ by setting $N = n$. This is not possible in case (ii), but is possible in case (i) by assuming, without loss of generality, that $r = x^2\ell$ for some $x \in \mathbb{N}$. □

Previously, the best low-error explicit extractors for total-entropy sources [CGGL20] required entropy $\Gamma \geq \max\{(r\ell)^{2/3+\delta}, r^{1/2+\delta}\ell\}$. Non-constructively, we know it is possible [KRVZ06] to achieve an entropy requirement of $\Gamma \geq O(\ell + \log r)$ and error of $2^{-\Omega(\Gamma)}$. Thus, while there is still a lot of room to give improved explicit extractors for total-entropy sources, we remark that the entropy requirement in Theorem 5.6 is, in fact, not far from optimal when $\ell \gg r$.

Finally, we show how to combine our improved explicit extractors for total-entropy sources (Theorem 5.6) with the standard reduction from small-space sources to total-entropy sources (Lemma 5.4) to complete the proof of Theorem 5.1:

Proof of Theorem 5.1. Fix any $\delta \in (0, 1/2]$, and let $\alpha \in (0, 1/4]$ be a sufficiently small constant and $C > 0$ a sufficiently large constant. Given a space s source $\mathbf{X}$ over $\{0,1\}^n$ with entropy $k \geq Cn^{1/2+\delta}s^{1/2-\delta}$, we know by Lemma 5.4 that $\mathbf{X}$ is $\epsilon_0 = 2^{-k/4}$-close to a convex combination of $(r, \ell, k/2)$-total entropy sources, where $r = \alpha k/s$ and $\ell = ns/(\alpha k)$. (Here we assume $r, \ell \in \mathbb{N}$, but it is easy to extend the argument when this is not the case.) In particular, this means there is some random variable $\mathbf{Y}$ such that with probability at least $1 - \epsilon_0$ over $y \sim \mathbf{Y}$, the random variable $(\mathbf{X} \mid \mathbf{Y} = y)$ is an $(r, \ell, k/2)$-total entropy source.

Let $\mathsf{Ext}_0 : (\{0,1\}^{\ell})^{r} \to \{0,1\}^{m}$ be the extractor from Theorem 5.6 for such total-entropy sources. We will argue that it also an extractor for the small-space source $\mathbf{X}$. Notice we have $|\mathsf{Ext}_0(\mathbf{X}) - \mathbf{U}_m| \leq \mathbb{E}_{y \sim \mathbf{Y}}[|\mathsf{Ext}_0(\mathbf{X} \mid \mathbf{Y} = y) - \mathbf{U}_m|] \leq \epsilon_0 + |\mathsf{Ext}_0(\mathbf{X}') - \mathbf{U}_m|$, where $\mathbf{X}'$ is some $(r, \ell, k/2)$-total entropy source. If we can argue that $r, \ell, k/2$ are sufficiently large and $k/2 \geq \max\{(r\ell)^{1/2+\delta}, r^{\delta}\ell\}$, then Theorem 5.6 tells us that $|\mathsf{Ext}_0(\mathbf{X}) - \mathbf{U}_m| \leq \epsilon_0 + |\mathsf{Ext}_0(\mathbf{X}') - \mathbf{U}_m| \leq 2^{-k/4} + 2^{-(r\ell)^{\Omega(1)}} = 2^{-n^{\Omega(1)}}$ and $m = (r\ell)^{\Omega(1)} = n^{\Omega(1)}$, which would prove the current theorem. We know that $r, \ell, k/2$ are sufficiently large because $r = \alpha k/s \geq \alpha C(n/s)^{1/2+\delta} \geq \alpha C$, and $\ell = ns/(\alpha k) \geq 1/\alpha$, and $k \geq C$, where α is sufficiently small and C is sufficiently large. Next, we know $k/2 \geq (r\ell)^{1/2+\delta} = n^{1/2+\delta}$ by the provided lower bound on k. Finally, to show $k/2 \geq r^{\delta}\ell = (\alpha k/s)^{\delta} ns/(\alpha k)$, rearrange the inequality to obtain $k^{2-\delta} \geq 2\alpha^{\delta-1}s^{1-\delta}n$, plug in the provided lower bound on k to obtain $(Cn^{1/2+\delta}s^{1/2-\delta})^{2-\delta} \geq 2\alpha^{\delta-1}s^{1-\delta}n$, and observe that it therefore suffices to show $(0.5C^{2-\delta}\alpha^{1-\delta}) \cdot n^{(1/2+\delta)(2-\delta)-1} \geq s^{1-\delta-(2-\delta)(1/2-\delta)}$, or rather

$$(0.5C^{2-\delta}\alpha^{1-\delta}) \cdot n^{2\delta-\delta/2-\delta^2} \geq s^{2\delta-\delta/2-\delta^2}.$$

This holds because $n \geq s$ (otherwise the provided lower bound on k gives $k > n$), because $2\delta - \delta/2 - \delta^2 \geq 0$ over $\delta \in (0, 1/2]$, and because C is sufficiently large. □

We conclude this section with a remark about the $\sqrt{n}$ "barrier" in small-space extraction.

Remark 5.7. *Any significant improvement to our extractors for small-space sources, presented in Theorem 5.1, would require significantly new techniques. In particular, if one wishes to construct extractors that can handle min-entropy $k \geq \sqrt{n}$, it is not possible to use the standard reduction from small-space sources to total-entropy sources appearing in the proof to Lemma 5.5. This is because if $k = \sqrt{n}$, then either $\ell > \sqrt{n}$, or not. If $\ell > \sqrt{n}$, then all the entropy could lie in one chunk of length $\ell > k$, and it is impossible to extract from one source. If $\ell \leq \sqrt{n}$, then $r \geq \sqrt{n}$, and the application of Lemma 2.3 in Equation (3) always leaves the source $(\mathbf{X} \mid \mathbf{W}^* = w)$ with 0 entropy, from which extraction is impossible.*

6 Future directions

In this paper, we give the first derandomization of Rödl and Šinajová's probabilistic designs [RŠ94], and show how they can be used to get better extractors for adversarial sources, total-entropy sources, and small-space sources. The three most natural open problems are as follows, which ask for improvements on each of our three main theorems Theorems 1 to 3, respectively.

Problem 1. *Better explicit designs: improve the constant in the power of n of Theorem 1 from 2 to 1.99. Alternatively, extend Theorem 1 to also work when r is odd.*

Problem 2. *Better extractors for adversarial sources: improve the requirement on good sources in Theorem 2 from $K \geq N^{\delta}$ to $K \geq N^{o(1)}$, or even $K \geq \operatorname{polylog} N$.*

Problem 3. *Better extractors for small-space sources: improve the requirement on min-entropy in Theorem 3 from $k \geq n^{1/2+\delta}$ to $k \geq n^{0.499}$.*

An answer to Problem 1 would provide explicit designs for regimes in which we currently only have trivial bounds (i.e., when r is odd or $r \geq 2s$), while an answer to Problem 2 would likely require extractors that have stronger conditioning properties than even our best leakage-resilient extractors. Finally, an answer to Problem 3 might require an idea that bypasses a reduction to total-entropy sources, as explained in Remark 5.7. In general, any significant progress on these problems should require significantly new techniques, which would be interesting in their own right.

References

[BACDTS19] Avraham Ben-Aroya, Gil Cohen, Dean Doron, and Amnon Ta-Shma. Two-source condensers with low error and small entropy gap via entropy-resilient functions. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2019)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2019.

[BRC60] Raj Chandra Bose and Dwijendra K. Ray-Chaudhuri. On a class of error correcting binary group codes. *Information and Control*, 3(1):68–79, 1960.

[CGG+20] Eshan Chattopadhyay, Jesse Goodman, Vipul Goyal, Ashutosh Kumar, Xin Li, Raghu Meka, and David Zuckerman. Extractors and secret sharing against bounded collusion protocols. In *Proceedings of the 61st Annual IEEE Symposium on Foundations of Computer Science (FOCS, to appear)*. IEEE, 2020.

[CGGL20] Eshan Chattopadhyay, Jesse Goodman, Vipul Goyal, and Xin Li. Extractors for adversarial sources via extremal hypergraphs. In *Proceedings of the 52nd Annual*

ACM SIGACT Symposium on Theory of Computing, STOC 2020, pages 1184–1197, New York, NY, USA, 2020. Association for Computing Machinery.

[CL16] Eshan Chattopadhyay and Xin Li. Extractors for sumset sources. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pages 299–311. ACM, 2016.

[Eus13] Alexander Eustis. *Hypergraph independence numbers.* PhD thesis, UC San Diego, 2013.

[EV13] Alex Eustis and Jacques Verstraëte. On the independence number of Steiner systems. *Combinatorics, Probability & Computing*, 22(2):241–252, 2013.

[GB10] Venkatesan Guruswami and Eric Blais. Notes 6: Reed-Solomon, BCH, Reed-Muller and concatenated codes. *Introduction to Coding Theory CMU: Spring*, 2010.

[GPR95] David A. Grable, Kevin T. Phelps, and Vojtěch Rödl. The minimum independence number for designs. *Combinatorica*, 15(2):175–185, 1995.

[Hoc59] Alexis Hocquenghem. Codes correcteurs d'erreurs. *Chiffres*, 2(2):147–56, 1959.

[KMS19] Ashutosh Kumar, Raghu Meka, and Amit Sahai. Leakage-resilient secret sharing against colluding parties. In *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 636–660. IEEE, 2019.

[KMV14] Alexandr Kostochka, Dhruv Mubayi, and Jacques Verstraëte. On independent sets in hypergraphs. *Random Structures & Algorithms*, 44(2):224–239, 2014.

[KRVZ06] Jesse Kamp, Anup Rao, Salil Vadhan, and David Zuckerman. Deterministic extractors for small-space sources. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, pages 691–700. ACM, 2006.

[Li15] Xin Li. Three-source extractors for polylogarithmic min-entropy. In *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*, pages 863–882. IEEE, 2015.

[MW97] Ueli Maurer and Stefan Wolf. Privacy amplification secure against active adversaries. In *Annual International Cryptology Conference*, pages 307–321. Springer, 1997.

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

[RŠ94] Vojtěch Rödl and Edita Šinajová. Note on independent sets in Steiner systems. *Random Structures & Algorithms*, 5(1):183–190, 1994.

[Sha11] Ronen Shaltiel. An introduction to randomness extractors. In *International Colloquium on Automata, Languages, and Programming*, pages 21–41. Springer, 2011.

[Sid95] Alexander Sidorenko. What we know and what we do not know about turán numbers. *Graphs and Combinatorics*, 11(2):179–199, 1995.

[Sid18] Alexander Sidorenko. Extremal problems on the hypercube and the codegree Turán density of complete r-graphs. *SIAM Journal on Discrete Mathematics*, 32(4):2667–2674, 2018.

[Sid20] Alexander Sidorenko. On generalized Erdős–Ginzburg–Ziv constants for $\mathbb{Z}_2^d$. *Journal of Combinatorial Theory, Series A*, 174:105254, 2020.

[Spe77] Joel Spencer. Asymptotic lower bounds for Ramsey functions. *Discrete Mathematics*, 20:69–76, 1977.

[TL18] Fang Tian and Zi-Long Liu. Bounding the independence number in some (n, k, ℓ, λ)-hypergraphs. *Graphs and Combinatorics*, 34(5):845–861, 2018.

[TV00] Luca Trevisan and Salil Vadhan. Extracting randomness from samplable distributions. In *Proceedings 41st Annual Symposium on Foundations of Computer Science*, pages 32–42. IEEE, 2000.

[Vad12] Salil Vadhan. Pseudorandomness. *Foundations and Trends® in Theoretical Computer Science*, 7(1–3):1–336, 2012.