Extracting Randomness via Repeated Condensing

Omer Reingold*
AT&T Labs - Research.

Ronen Shaltiel†
Hebrew University and IAS

Avi Wigderson‡
Hebrew University and IAS

Abstract

On an input probability distribution with some (min-)entropy an extractor *outputs a distribution with a (near) maximum entropy rate (namely the uniform distribution). A natural weakening of this concept is a* condenser, *whose output distribution has a higher entropy rate than the input distribution (without losing much of the initial entropy).*

In this paper we construct efficient explicit condensers. The condenser constructions combine (variants or more efficient versions of) ideas from several works, including the block extraction scheme of [10], the observation made in [15, 9] that a failure of the block extraction scheme is also useful, the recursive "win-win" case analysis of [4, 5], and the error correction of random sources used in [17].

As a natural byproduct, (via repeated iterating of condensers), we obtain new extractor constructions. The new extractors give significant qualitative improvements over previous ones for sources of arbitrary min-entropy; they are nearly optimal simultaneously *in the main two parameters - seed length and output length. Specifically, our extractors can make any of these two parameters optimal (up to a constant factor), only at a* poly-logarithmic *loss in the other. Previous constructions require* polynomial *loss in both cases for general sources.*

We also give a simple reduction converting "standard" extractors (which are good for an average seed) to "strong" ones (which are good for most seeds), with essentially the same parameters. With it, all the above improvements apply to strong extractors as well.

*AT&T Labs - Research. Room A243, 180 Park Avenue, Bldg. 103, Florham Park, NJ, 07932, USA. E-mail: omer@research.att.com. Part of this research was performed while visiting the Institute for Advanced Study, Princeton, NJ.

†Department of Computer Science, Hebrew University, Jerusalem, Israel, and Institute for advanced study, Princeton, NJ. E-mail:ronens@cs.huji.ac.il.

‡Department of Computer Science, Hebrew University, Jerusalem, Israel, and Institute for advanced study, Princeton, NJ. E-mail:avi@ias.edu.

This research was supported by USA-Israel BSF Grant 97-00188.

1. Introduction

1.1. Extractors

A line of research, (initiated by [18, 1, 14]) is motivated by the question of availability of truly random bits. The idea is to make truly random bits available (for example to probabilistic algorithms) by refining the (imperfect) randomness found in some natural physical processes. It was shown by [14] that this task cannot be performed by deterministic algorithms, even when assuming that the source has some "nice" structure. In light of this, the goal of this line of research became "spending" as few as possible truly random bits in order to extract as many as possible (almost) truly random bits from arbitrary imperfect random sources which contain sufficient randomness.

The most general definition of weak random sources and the formal definition of extractors emerged from the works of [20, 21, 10]. The definition of extractors requires quantifying two notions: The first is the amount of randomness in probability distributions, which is measured using a variant of the entropy function called *min-entropy*.

Definition 1 *A distribution X is called a k-*source *if the probability it assigns to every element in its range is bounded by 2^{-k}. The* min-entropy *of X, (denoted by $H_\infty(X)$) is the maximal k such that X is a k-source.*

The second notion is the quality of the extracted bits, which is measured using the statistical distance between the extracted bits and truly uniform ones.

Definition 2 *Two distributions P, Q, (over the same domain Ω) are ϵ-*close *if they have statistical distance of at most ϵ. (For every event $A \subseteq \Omega$, the probability that the two distributions assign to A differ by at most ϵ). We use U_l to denote the uniform distribution on l bit strings.*

Extractors are functions which use *few* truly random bits to extract *many* (almost) truly random bits from arbitrary distributions which "contain" sufficient randomness.

Definition 3 *[10] A function $Ext : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is a (k, ϵ)-*extractor *if for every k-source X, the* *distribution $Ext(X, U_d)$, (obtained by running the extractor on an element sampled from X and a uniformly chosen d bit string, which we call the seed) is ϵ-close to U_m.*

Informally, we will say that an extractor uses a seed of length d in order to extract m bits from distributions on n bits which contain k random bits. We refer to the ratio between m and k as the fraction of the randomness which the extractor extracts, and to the ratio between k and n as the entropy rate of the source.

Apart from their obvious application, extractors turned out to be useful in many areas in complexity theory and combinatorics, with examples being pseudo-random generators for space bounded computation, deterministic amplification, oblivious sampling, constructive leader election and explicit constructions of expander graphs, superconcentrators and sorting networks. The reader is referred to the excellent survey papers [8, 9].

1.2. Previous Results

Extractor constructions are measured by viewing d and m as functions of the source parameters (n and k) and the required error ϵ. A recent result of [12] enables us to rid ourselves of ϵ, and assume that ϵ is a small constant[1]. We maintain this convention throughout the introduction.

When constructing extractors there are two possible objectives: minimizing the seed length d and maximizing the output size m. It should be noted that the existence of an optimal extractor (which optimizes both parameters simultaneously, and matches the known lower bounds) can be easily proven using the probabilistic method. Thus, the goal is to match this performance with explicit constructions[2].

In the remainder of this sub-section we survey the best explicit extractors constructions known for the two objectives. Tables 1,2 state more extractor constructions, but are far from covering the mass of work done in this area.

1. Minimizing the seed length: A lower bound of $\Omega(\log n)$ on the seed length was given in [10]. In contrast to the (non-explicit) optimal extractor which uses a seed of length $O(\log n)$ to extract *all* the randomness from the source, explicit constructions of extractors can optimize the seed length only at the cost of extracting a small fraction of the randomness. The situation is not so bad for large k, ($k = \Omega(n)$) as the extractor of [22] uses a seed of length $O(\log n)$ to extract a constant fraction of the initial randomness. However, for smaller k, explicit constructions can only extract a polynomial fraction of the initial randomness, ($m = k^{1-\alpha}$ for an arbitrary constant α). This result was first achieved in [17] for $k = n^{\Omega(1)}$, and later extended for any k in [5].

Table 1. Extracting a constant fraction: ($m = (1-\alpha)k$ for arbitrary $\alpha > 0$)

reference	min-entropy k	seed length d
[22]	$k = \Omega(n)$	$O(\log n)$
[16]	any k	$O(\log^9 n)$
[5]	$k = 2^{O(\sqrt{\log n})}$	$O(\log n \cdot \log\log\log n)$
[13]	any k	$O(\log^2 n)$
Thm. 1	any k	$O(\log n \cdot (\log\log n)^2)$
optimal	any k	$O(\log n)$

Table 2. Optimizing the seed length: ($d = O(\log n)$)

reference	min-entropy k	output length m
[22]	$k = \Omega(n)$	$(1-\alpha)k$
[17]	$k = n^{\Omega(1)}$	$k^{1-\alpha}$
[5]	$k = 2^{O(\sqrt{\log n})}$	$\Omega(\frac{k}{\log\log\log n})$
[5]	any k	$k^{1-\alpha}$
Thm. 2	any k	$\Omega(\frac{k}{\log n})$
optimal	any k	k

All the results are stated for constant ϵ. α is an arbitrary small constant.

2. Maximizing the output size: Explicit extractors can extract large fractions of the randomness only at the cost of enlarging the seed length. A general method to increase the fraction extracted at the cost of enlarging the seed length was given in [19]. The best extractors which extract *all* the randomness out of random sources are constructed this way from extractors which extract a constant fraction of the randomness. In light of this, we focus our attention to extractors which extract a constant fraction. The best such explicit extractor is by [13], which uses a seed of length $O(\log^2 n)$.

Concluding this presentation, we stress that while there are explicit constructions which are optimal in any of these two parameters, the cost is a polynomial loss in the other.

1.3. New Results

We give two constructions, each optimal in one of the parameters and losing only a poly-logarithmic factor in the other. Thus, both come closer to simultaneously optimizing both parameters. The results are stated for constant ϵ, (see section 7 for the exact dependence on ϵ). In the first construction we extract any constant fraction of the initial randomness using a seed of length $O(\log n \cdot (\log\log n)^2)$. This improves the best previous such result by [13] which uses a seed of length $O(\log^2 n)$.

[1] [12] gave a general explicit transformation which transforms an extractor with constant error into an extractor with arbitrary small error while harming the other parameters only slightly more than they need to be harmed. The exact dependence of our results on ϵ is presented in section 7.

[2] A family of extractors is explicit if it can be computed in polynomial time. Formally, a family $E = \{E_n\}$ of extractors is defined given polynomially computable integer functions $d(n), m(n), k(n), \epsilon(n)$ where $E_n : \{0,1\}^n \times \{0,1\}^{d(n)} \to \{0,1\}^{m(n)}$ is a $(k(n), \epsilon(n))$-extractor. The family is explicit in the sense that E_n can be computed in time polynomial in n.

Theorem 1 *For every n, k and constant ϵ, there are explicit (k,ϵ)-extractors $Ext : \{0,1\}^n \times \{0,1\}^{O(\log n \cdot (\log\log n)^2)} \rightarrow \{0,1\}^{(1-\alpha)k}$, where $\alpha > 0$ is an arbitrary constant.*

Using [19], we get the following corollary[3], which also improves the previous best construction which extract all the randomness by [13].

Corollary 1 *For every n, k and constant ϵ, there are explicit (k,ϵ)-extractors $Ext : \{0,1\}^n \times \{0,1\}^{O(\log n \cdot (\log\log n)^2 \cdot \log k)} \rightarrow \{0,1\}^k$.*

Our second construction uses the optimal seed length (that is $O(\log n)$) to extract $m = \Omega(k/\log n)$ bits, this improves the best previous result by [5] which could only extract $m = k^{1-\alpha}$ bits.

Theorem 2 *For every n, k and constant ϵ, there are explicit (k,ϵ)-extractors $Ext : \{0,1\}^n \times \{0,1\}^{O(\log n)} \rightarrow \{0,1\}^{\Omega(k/\log n)}$.*

Using [5], we get the following corollary, (in which the "loss" depends only on k)[4].

Corollary 2 *For every n, k and constant ϵ, there are explicit (k,ϵ)-extractors $Ext : \{0,1\}^n \times \{0,1\}^{O(\log n)} \rightarrow \{0,1\}^{\Omega(k/(\log k \cdot \log\log k))}$.*

Condensers

The main construction of this paper is of a primitive we call a condenser (which is a natural weakening of an extractor). We call the ratio between the amount of randomness in a source and its length the *entropy-rate* of the source. Using this terminology, an extractor is a function that uses few random bits to convert any random source which contains enough randomness into a source with entropy rate 1. A condenser[5] is a function that uses few random bits to convert any random source which contains enough randomness into a source of larger entropy rate.

Though weaker than extractors, condensers can be used to construct extractors by repeatedly condensing the source until its entropy-rate becomes close to 1.

Transforming general extractors into strong extractors

Speaking informally, a strong extractor is an extractor in which the output distribution is independent of the seed[6]. In some applications of extractors it is beneficial to have the strong version. Most extractor constructions naturally lead to strong extractors, yet some (with examples being [16, 5] and the constructions of this paper), are not strong or difficult to analyze. We solve this difficulty by giving a general explicit transformation which transforms any extractor into a strong extractor with essentially the same parameters. Exact details will appear in the full version.

1.4. Technique

High level overview

In contrast to latest extractors papers [17, 13, 5] we do not use Trevisan's paradigm. Instead, we revisit [10] and attempt to construct block sources, (A special kind of sources which allow very efficient extraction). Following [15, 9], we observe that when failing to produce a block source the method of [10] "condenses" the source. This enables us to use a "win-win" case analysis as in [4, 5] which eventually results in a construction of a condenser. Our extractors are then constructed by repeated condensing. The parameters of these extractors are not as good as those stated in section 1.3. In order to get the promised parameters, we improve the construction of [10].

Block sources

One scenario in which very efficient extraction is possible, is when the source distribution X is made of two independent concatenated distributions $X = (X_1, X_2)$, where X_1 is a k_1-source and X_2 is a k_2-source. Extractors for this special scenario (which are called block-source extractors) can be constructed by composing two extractors: An extractor with optimal seed length can be used to extract random bits from X_2, and these bits, (being independent of X_1), can be used as seed to extract all the randomness from X_1 using an extractor with large output. (Note, that with today's best extractors this argument uses the optimal seed length to extract all the randomness from X_1, as long as k_2 is at least $polylog(n)$). The requirement that X_1 and X_2 be independent could be relaxed in the following way, (that was suggested in [2]): Intuitively, it is sufficient that X_1 contains k_1 random bits, and X_2 contains k_2 random bits which are not contained in X_1. Such sources are called block-sources[7]. Thus, extracting randomness from general sources can be achieved by giving a construction which uses few random bits to transform a general source into a block source, and using a block-source extractor.

This approach was suggested by Nisan and Zuckerman in [10]. They constructed a "block extraction scheme", that is a scheme which given an arbitrary source X, uses few random bits to produce a new source B (called a block) which

[3]The corollary does not follow directly from the version of theorem 1 stated here. A stronger version in which $\epsilon = 1/polylog(n)$ is necessary. The stronger version follows from our exact analysis, see section 7.

[4][5] show that an extractor $Ext : \{0,1\}^{k^{O(1)}} \times \{0,1\}^d \rightarrow \{0,1\}^m$ can be used to construct an extractor $Ext : \{0,1\}^n \times \{0,1\}^{d+O(\log n)} \rightarrow \{0,1\}^{\Omega(m/\log\log k)}$ for any n.

[5]The notion of condensers was also used in [11]. While similar in spirit, that paper deals with a completely different set of parameters and uses different techniques.

[6]In the original paper [10] strong versions of extractors were defined and constructed, and the notion of a "non-strong" extractor was later given by Ta-Shma, [16].

[7]More precisely, a (k_1, k_2)-block source is a distribution $X = (X_1, X_2)$ such that X_1 is a k_1-source, and for every possible value x_1 of X_1, the distribution of X_2, conditioned on the event that $X_1 = x_1$, is a k_2-source.

is shorter than the initial source, and contains a large fraction of the initial randomness. This means that the distribution (B, X) meets the first requirement of block sources: The first block contains randomness. Intuitively, to meet the second requirement one should give an upper bound on the amount of randomness contained in B, and conclude that there is some randomness in X which is not contained in B. However, in the construction of Nisan and Zuckerman such an upper bound can be achieved only "artificially", by making the length of B smaller than k. This has a costly effect, as the amount of randomness that is guaranteed to be in B is proportional to its length. In particular, when $k < \sqrt{n}$ choosing the length of B this way, it may be the case that B contains no randomness. As a result, the extractors of [10] do not work when $k < \sqrt{n}$.

A "win-win" analysis

A way to go around this problem was suggested in [15, 9]. The idea is to argue that when the block extraction scheme fails to produce a block source, it produces a block B which is more "condensed" than the initial source. More precisely, we will use the block extraction scheme to produce a block B, (say of length $n/2$). Consider the distribution (B, X). Recall that for our purposes, it suffices that X contains very few random bits that are not contained in B. In other words, when we fail to construct a block source, this happens because (almost) all the randomness "landed" in B. In this case, we obtained a block that is more condensed than the initial source X. (It has roughly the same amount of randomness and half the length).

Using this idea recursively, at each step either we construct a block source, (from which we can extract randomness), or we condense the source. There is a limit on how much we can condense the source. Certainly, when the length reduces to k, no further condensing is possible. This means that running this procedure recursively enough times we will obtain a block source.

The outcome of the above procedure is several "candidate" distributions, where one of them is a block source. Not knowing which is the "right one", we run block source extractors on all of them, (using the same seed). We know that one of the distributions we obtain is close to uniform. It turns out that the number candidate distributions is relatively small. This means that concatenating the output distributions produces a new source which is much shorter than the initial source and contains a large fraction of the initial randomness. Thus, we have constructed a condenser. Our extractors are then constructed by repeatedly condensing the source until it becomes close to uniform.

A very similar approach has already been used in [9]. (Actually, the construction in that paper is far more complex, partly because the extractors they had available were not as good as the ones we have now). Still, (even when plugging in very good extractors), the method explained above is too costly and ends up constructing an extractor that uses a *large* seed and extracts a *small* fraction of the randomness. We overcome this problem by improving the main primitive in our construction: the block extraction scheme of Nisan and Zuckerman.

Improved block extraction

Let us delve into the parameters. The block extraction scheme of Nisan and Zuckerman spends $O(\log n)$ random bits when producing a block B of length $n/2$, and can guarantee that B is an $\Omega(k/\log(n/k))$-source. This turns out to be too costly to run our recursive construction.

One problem is that the number of random bits used by the block extraction scheme is too large for our purposes. Since the block extraction scheme already spends $O(\log n)$ random bits, we can only afford to run it a constant number of times if we want to shoot for optimal seed length. Using the strategy described above we will need to run it roughly $\log n$ times, resulting in a large seed length. We overcome this problem by derandomizing the construction of Nisan and Zuckerman. We reduce the number of random bits used from $\log n$ to $\log \log n$, allowing us to run it a larger number of times. We find it surprising that some non-trivial extraction task can be performed using a sub-logarithmic amount of random bits.

A second problem is that we want the block B to contain a constant fraction of the initial randomness. In the construction of Nisan and Zuckerman this happens only when $k = \Omega(n)$. To overcome this problem we show how to reduce the case of $k = o(n)$ into the case of $k = \Omega(n)$. This is done by error correcting the source prior to using the block extraction scheme. We give a non-constructive argument to show that every error corrected random source has a "piece" of length k which is an $\Omega(k)$-source. Intuitively, this enables the block extraction scheme to be carried out on the condensed piece, where it performs best[8].

1.5. Organization of the paper

In section 2 we define block sources. In section 3 we construct a block extraction scheme. In section 4 we use the method of [9] to show that when using the block extraction scheme either we get a block source or we condense the source. In section 5 we run the block extraction scheme recursively and obtain condensers. In section 6 we use the condensers to construct extractors. Section 7 gives the exact dependence of our extractors on the error parameter ϵ.

2. Block sources

Block sources are random sources which have a special structure. The notion of block sources was defined in [2].

[8]As far as we know, the idea of error correcting random sources (in the context of extracting randomness) first appeared in [17]. Interestingly, we use error correcting codes with slightly different properties, and use a different method in analyzing them.

Definition 4 *[2] Two random variables (X_1, X_2) form a (k_1, k_2)-block source if X_1 is a k_1-source, and for every possible value x_1 of X_1 the distribution of X_2, given that $X_1 = x_1$, is a k_2-source.*

Block source extractors are extractors which work on block sources.

Definition 5 *A (k, t, ϵ)-block source extractor is a function $Ext : \{0,1\}^{n_1} \times \{0,1\}^{n_2} \times \{0,1\}^d \to \{0,1\}^m$, such that for every (k, t)-block source (X_1, X_2), (where the X's are of length n_1,n_2 respectively), the distribution $Ext(X_1, X_2, U_d)$ is ϵ-close to U_m.*

The definition of block sources is tailored to allow the following composition of extractors.

Theorem 3 *(implicit in [10]) If there exist an explicit (k, ϵ_1)-extractor $Ext_1 : \{0,1\}^{n_1} \times \{0,1\}^{d_1} \to \{0,1\}^m$, and an explicit (t, ϵ_2)-extractor $Ext_2 : \{0,1\}^{n_2} \times \{0,1\}^{d_2} \to \{0,1\}^{d_1}$, then there exists an explicit $(k, t, \epsilon_1 + \epsilon_2)$-block-source extractor $Ext : \{0,1\}^{n_1} \times \{0,1\}^{n_2} \times \{0,1\}^{d_2} \to \{0,1\}^m$.*

We can use the above theorem to compose two extractors: one which optimizes the seed length and another which optimizes the output length. The resulting block-source extractor will "inherit" the nice properties of both its component extractors. Particularly, taking Ext_1 to be the extractor of [13] and Ext_2 to be the extractor of [5], we get the following block-source extractor:

Corollary 3 *For every n_1, n_2, k and $t \geq \log^4 n_1$ there exists an explicit $(k, t, \frac{1}{n_1} + \frac{1}{n_2})$-block source extractor $BE : \{0,1\}^{n_1} \times \{0,1\}^{n_2} \times \{0,1\}^{O(\log n_2)} \to \{0,1\}^k$.*

Remark 1 *In the rest of the paper we assume that $k \geq \log^4 n$. We need need this assumption since we're going to transform a k-source into a block-source in which the second block contains $\log^4 n$ bits. We can make this assumption because for $k = 2^{O(\sqrt{\log n})}$, theorems 1,2 follow from the constructions in [5], (see tables 1,2).*

Thus, to construct extractors which achieve short seed length and large output simultaneously, it suffices to use few random bits, and convert any k-source into a $(k', \log^4 n)$-block source such that k' is not much smaller than k.

This turns out to be a tricky problem, no such (efficient in terms of random bits spent) scheme is known when $k < \sqrt{n}$.

3. Improved block extraction

Current constructions of block sources rely on a building block which we will call a "block extraction scheme". Intuitively, this is a scheme which given an arbitrary source X, uses few random bits to produce a new source B (called a block) which is shorter than the initial source, and contains a large fraction of the initial randomness. As explained in the introduction, given such a scheme, we may hope that B does not contain all the randomness of X, which intuitively means that the distribution (B, X) is a block source.

The following definition formalizes the notion of a block extraction scheme.

Definition 6 *A $(n, k, u, r, \gamma, \rho)$-block extraction scheme is a function:*

$$B : \{0,1\}^n \times \{0,1\}^u \to \{0,1\}^{\frac{n}{r}}$$

Such that for every k-source X on $\{0,1\}^n$, the distribution $B(X, Y)$, (obtained by sampling X from the source, and y uniformly from $\{0,1\}^u$), is ρ-close to a $\frac{\gamma k}{r}$-source.

Intuitively, the block extraction scheme spends u random bits in order to produce a block B of length n/r, which is guaranteed to be ρ-close to a $\gamma k/r$-source[9].

Using this notation, Nisan and Zuckerman proved the following theorem:

Lemma 1 *[10] There exists an explicit $(n, k, O(\log n \log \frac{1}{\rho}), r, \Omega(\frac{1}{\log \frac{n}{k}}), \rho)$-block extraction scheme, as long as $\rho \geq 2^{-ck}$ for some constant c.*

The following lemma improves lemma 1.

Lemma 2 *There exists an explicit $(n, k, O(\log \log n + \log(1/\rho) + \log r), r, \Omega(1), \rho)$-block extraction scheme as long as $\rho \geq c\sqrt{\frac{r}{k}}$ for some constant c.*

Lemma 2 improves lemma 1 in two respects (as long as one settles for small r and large ρ).

1. We reduce the number of random bits spent by the block extraction scheme. In [10] the number of random bits is logarithmic in n, whereas in lemma 2 the number of random bits is double logarithmic in n.

   This is achieved by derandomizing the proof of Nisan and Zuckerman using small bias spaces. In section 3.2 we describe the property that a distribution needs to have in order to allow the Nisan-Zuckerman analysis, and construct a small sample space with this property.

2. We increase the amount of randomness guaranteed in the output block. In [10] the amount of randomness guaranteed in the output block B is $\Omega(\frac{k}{r \log(n/k)})$. Lemma 2 guarantees that B contain $\Omega(\frac{k}{r})$ random bits.

   Note that the two quantities are the same when $k = \Omega(n)$. Indeed, our improvement is achieved by reducing the case of $k = o(n)$ to that of $k = \Omega(n)$. In section 3.1 we show that once a random source is error

[9] The block extraction scheme of [10] has a stronger property, namely that for a $(1-\rho)$-fraction of the y's, $B(X, y)$ is ρ-close to a $\frac{\gamma k}{r}$-source. This property is also shared by our new construction, but is not needed later in the analysis. Another interesting property of both constructions is that they do not depend on k.

corrected, there are some k indices, (to the error corrected source) which induce an $\Omega(k)$-source. Thus, by error correcting the source we can focus on a small portion of it in which the ratio between min-entropy and length is constant. The argument of Nisan and Zuckerman is then carried out only on this portion of the source where it performs best. The exact analysis is given in section 3.3.

3.1. Error corrected random sources

In this subsection we show that if we apply an error correcting code to an arbitrary k-source, we obtain a k-source which has k indices which induce an $\Omega(k)$-source.

We use the following construction of error correcting codes.

Theorem 4 *[6] There exist constants a, b and an explicit error correcting code $EC : \{0,1\}^n \to \{0,1\}^{an}$ such that for every $x_1 \neq x_2 \in \{0,1\}^n$, $d(EC(x_1), EC(x_2)) \geq bn$. ($d(z_1, z_2)$ denotes the Hamming distance between z_1, z_2).*

In the remainder of this section we fix a, b and EC to be these of theorem 4. For a vector $x \in \{0,1\}^n$ and a set $T \subseteq [n]$ we use $x|_T$ to denote the restriction of x to T.

The following lemma was mentioned to us by Russell Impagliazzo and seems to be folklore.

Lemma 3 *[3] Let X be a k-source on $\{0,1\}^n$. There exists a set $T \subseteq [an]$ of size k, such that $EC(X)|_T$ is an $\Omega(k)$-source.*

The proof of lemma 3 will be given in the full version,

3.2. A small sample space for intersecting large sets

The block extraction scheme of Nisan and Zuckerman is obtained by restricting the source to some subset of the indices which is selected using few random bits. More precisely, they construct a small sample space of small subsets of $[n]$ (having a property that we immediately describe) and prove that the distribution obtained by sampling an element from a k-source and restricting it to the indices in a random set from the sample space contains a large fraction of the initial randomness. In this section we construct a smaller such sample space which enables us to spend less random bits to construct a block extraction scheme.

It will be convenient to have a notion of efficiently constructible distributions.

Definition 7 *Call a distribution P on n bits, polynomially constructible using $u(n)$ bits[10], if there exists an algorithm $A : \{0,1\}^{u(n)} \to \{0,1\}^n$ which runs in time polynomial in n, such that the distribution $A(Y)$ where Y is chosen uniformly from $\{0,1\}^{u(n)}$ is identical to P.*

[10]Naturally, one should speak about a sequence $P = \{P_n\}$ of distributions for this to make sense.

Intuitively, a k-source has k random bits "hidden" somewhere. This suggests that the property needed is that the distribution of sets will "intersect" any set of k indices.

Definition 8 *A distribution S over subsets of $[n]$ is called (n, k, r, δ)-intersecting if for every $G \subseteq [n]$ with $|G| \geq k$, $\Pr_S(|S \cap G| < \frac{k}{8r}) < \delta$.*

The following is implicit in [10][11].

Lemma 4 *[10] There exists some constant c such that if X is a k-source on $\{0,1\}^n$ and S is a distribution over subsets of $[n]$ which is $(n, \frac{ck}{\log(n/k)}, r, \delta)$-intersecting then the distribution $X|_S$ (obtained by sampling x from X and s from S and computing $x|_s$) is $(4\sqrt{\delta} + 2^{-\Omega(k)})$-close to a $\Omega(\frac{k}{r\log(n/k)})$-source.*

Nisan and Zuckerman use a construction based on $O(\log(1/\delta))$-wise independence to prove the following lemma.

Lemma 5 *[10] For every n, k, r and $\delta > 2^{-O(k/r)}$ there is a (n, k, r, δ)-intersecting distribution on subsets of size n/r which is polynomially constructible using $O(\log n \cdot \log(1/\delta))$ bits.*

Using lemma 4 this immediately implies the block extraction scheme of lemma 1. We will be mostly interested in the case when r is small, (say $r \leq \log n$) and δ is large, (say $\delta \geq \log^{-O(1)} n$). For this setup we can save random bits and make the dependence on n double logarithmic.

Lemma 6 *There exists a constant c such that for every n, k, r and $\delta > cr/k$, there is a (n, k, r, δ)-intersecting distribution on subsets of size n/r which is polynomially constructible using $O(\log\log n + \log r + \log(1/\delta))$ bits.*

In the rest of this section we prove lemma 6. We will view distributions over n bit strings as distributions over subsets of $[n]$. (Using the standard correspondence $(W_1, \cdots, W_n) \leftrightarrow \{i | W_i = 1\}$). We will construct a distribution $(W_1, \cdots, W_n)$ with the following properties:

- For every $1 \leq i \leq n$, $\Pr(W_i = 1) \approx 1/2r$.
- For every set $G \subseteq [n]$ with $|G| \geq k$, the probability that the sum of the W_i's for $i \in G$ is far from the expected $|G|/2r$ is small. (It is important to note that we allow the "small probability" to be quite large, since we are shooting for large δ).

Note that the second condition gives both the intersecting property and the fact that the selected sets are rarely of size larger than n/r. We are interested in constructing such a distribution using as few as possible random bits. A pairwise

[11]It should be noted that the argument of [10] is not as straightforward as the above intuition suggests. In fact, the proof is quite involved and a hint to that may be found in the fact that the distribution needs to intersect sets of size $\frac{k}{\log(n/k)}$ and not k.

independent distribution has these properties but takes $\log n$ random bits to construct. We can do better by using the "almost l-wise dependent" distributions of [7]. A technicality is that we want a distribution in which the probability that a single bit evaluates to 1 is close to $1/2$ and not to $1/2r$.

Let us define almost l-wise dependent distributions.

Definition 9 *([7]) A distribution $(P_1, \cdots, P_n)$ over $\{0,1\}^n$ is said to be (ϵ, l)-wise dependent with mean p if for every subset $\{i_1, \cdots, i_l\}$ of $[n]$, the distribution $(P_{i_1}, \cdots, P_{i_l})$ is ϵ-close to the distribution over l bit strings where all bits are independent and each of them takes the value 1 with probability p.*

Naor and Naor showed that almost l-wise dependent distributions can be constructed using very few random bits.

Theorem 5 *[7] For every n, l and ϵ, an (ϵ, l)-wise dependent distribution with mean $1/2$ is polynomially constructible using $O(\log\log n + l + \log(1/\epsilon))$ bits.*

The following corollary follows easily by taking an almost $2q$-dependent distribution over qn bits, (with mean close to $1/2$), viewing it as n blocks of q bits, and setting the variable of the i'th block to the value 1 if all the variables in the corresponding block are 1.

Corollary 4 *For every n, ϵ and q, an $(\epsilon, 2)$-wise dependent distribution with mean 2^{-q} is polynomially constructible using $O(\log\log n + q + \log(1/\epsilon))$ bits.*

We are ready to construct our intersecting distribution.

Definition 10 *Let q be an integer such that $1/4r < 2^{-q} \leq 1/2r$, and $\epsilon = min(c\delta^2, c/r^2)$ where c is a constant which will be determined later. Let $W = (W_1, \cdots, W_n)$ be the $(\epsilon, 2)$-wise dependent distribution with mean 2^{-q} guaranteed in corollary 4.*

Almost pairwise dependent distributions are very much like pairwise independent distributions in the sense that they allow the use of Chebychev's inequality.

Lemma 7 *[7] If $(X_1, \cdots, X_n)$ is a $(\epsilon, 2)$-wise dependent distribution with mean p, then for every $0 < \lambda < 1$*

$$\Pr(|\sum_{1 \leq i \leq n} X_i - pn| > \lambda pn) < O(\frac{1}{\lambda^2 pn} + \sqrt{\epsilon})$$

as long as $\epsilon < \frac{\lambda^2 p^2}{4}$.

The next lemma follows:

Lemma 8 *The constant c in definition 10 can be fixed in a way in which the distribution W has the following properties:*

1. *For every set $G \subseteq [n]$ such that $|G| \geq k$, $\Pr(\sum_{i \in G} W_i < \frac{k}{8r}) < \delta/3$.*
2. $\Pr(\sum_{1 \leq i \leq n} W_i > \frac{n}{r}) < \delta/3$.

Proof: We use lemma 7 to deduce both parts of the lemma. To meet the condition in lemma 7 we need to make sure that $\epsilon < \frac{\lambda^2 p^2}{4} = \Theta(1/r^2)$. The requirement that $\epsilon < c/r^2$ takes care of this condition for small enough constant c. Applying lemma 7 we get that the probability of deviation from the mean (in both parts of the lemma) is bounded by $O(r/k + \delta\sqrt{c}) = O(\delta\sqrt{c})$. (The last equality follows from the condition of lemma 6). This is bounded from above by $\delta/3$ for small enough c. •

We are ready to prove lemma 6

Proof: (of lemma 6) The first item of lemma 8 shows that W is $(n, k, r, \delta/3)$-intersecting, and the second item shows that W could be transformed into a distribution over subsets of size exactly n/r without changing it by much. This change is done by adding arbitrary indices to the set if its size is smaller than n/r and deleting arbitrary indices if its size is larger than n/r. Adding indices will not spoil the intersecting property, and the probability that we need to delete indices is bounded by $\delta/3$. The lemma follows. •

3.3. Construction of block extraction scheme

In this subsection we put everything together and prove lemma 2. We are ready to define our block extraction scheme.

Definition 11 *(block extraction scheme) Given r, ρ and a constant e (which will be fixed later) let $u = O(\log\log n + \log r + \log(1/\rho))$ be the number of bits used by lemma 6 to construct a $(an, ek, ar, (\frac{\rho}{4})^2)$-intersecting distribution. For $y \in \{0,1\}^u$, let S_y be the set defined by y in the intersecting distribution. We now define:*

$$B(x, y) = EC(x)|_{S_y}$$

We are finally ready to prove lemma 2.

Proof: (of lemma 2) Let V denote the distribution $EC(X)$. Lemma 3 implies that there exists a set $T \subseteq [an]$ of size k such that $V|_T$ is a dk-source, (for some constant d). Consider the distribution $S \cap T$, (the restriction of the intersecting distribution to the coordinates of T). It is trivial (from the definition) that this distribution is $(k, ek, ar, (\frac{\rho}{4})^2)$-intersecting. We now use lemma 4 on $V|_T$. Let us first check that the conditions of lemma 4 are met. We fix the constant e of definition 11, setting $e = (cd)/(-\log d)$, where c is the constant from lemma 4. The conditions of lemma 4 are met since $V|_T$ is a dk-source of length k and we have a distribution which is intersecting sets of size $ek = \frac{c(dk)}{\log(k/(dk))}$. We conclude that $V|_{S\cap T}$ is ρ-close to an $\Omega(k/r)$-source. Revealing more indices does not decrease the amount of randomness and thus $B(X,Y) = V|_S$ is ρ-close to an $\Omega(k/r)$-source. •

4. Partitioning to two "good" cases

In this section we show that using the block extraction scheme B (defined in section 3), essentially one of two "good" things happen. Either $(B(X,\cdot), X)$ is a block source or $B(X,\cdot)$ is much more condensed than X. Unfortunately, the above claim is not true. For some sources it may be the case that none of the two "good" cases applies[12]. However, this problem has already been solved in [9]. The solution is to prove a weaker statement which has the same flavor, and can be used the same way later on. The exact statement is that every source can be partitioned into three sets: The first has negligible weight and can be ignored. On the second, the block extraction scheme produces a block source, and on the third, the block extraction scheme condenses the source. To make this formal, we introduce the following notation:

Definition 12 *For a distribution P over a set Ω and $A \subseteq \Omega$ such that $P(A) \neq 0$, we define the conditional distribution P_A on Ω which gives weight $P(\omega)/P(A)$ for $\omega \in A$, and zero otherwise.*

We say that a random variable U on Ω, is a k-source in A, if the distribution $U(\omega)$, where ω is chosen according to P_A, is a k-source.

We say that two random variables U, V on Ω form a (k_1, k_2)-block source in A, if the distribution $(U(\omega), V(\omega))$, where ω is chosen according to P_A form a (k_1, k_2)-block source.

Recall that the current setup is the following: We have a k-source X on $\{0,1\}^n$, and an explicit $(n, k, u, r, \gamma, \rho)$-block extraction scheme B, (guaranteed from lemma 2 for some constant γ). For the remainder of this section we fix $t = \log^4 n$. (t is the amount of randomness needed in the second block of a block source in the block source extractor of corollary 3). We are ready to state the main lemma of this section.

Lemma 9 *(Following [9]) There exist a partition of $\{0,1\}^n \times \{0,1\}^u$ into three sets BAD, BLK, CON with the following properties:*

1. $\Pr(BAD) \leq 2(\rho + 2^{-t})$
2. *(B, X) is a $(\frac{\gamma k}{r} - t, t)$-block source in BLK.*
3. *B is a $(k - 2t)$-source in CON.*

The proof of lemma 9 will be given in the full version.

[12] An example is a source which flips a bit b and depending on the outcome decides weather to sample from a distribution in which the first k bits are random and the remaining $n - k$ bits are fixed, or from a k-wise independent distribution.

5. Constructing condensers

Condensers are a generalization of extractors. They are functions which use few random bits to "condense" an arbitrary random source. In this section we define and construct condensers. In 5.1 we use lemma 9 recursively to construct few distributions such that one of them is a block source. In 5.2 we show that in such a case applying block source extractors to the "candidate" distributions results in a condenser.

5.1. Getting a block source

In this subsection we implement the idea presented in the introduction. Namely, that running the block extraction scheme recursively, will eventually produce a block source. (The intuition is that in every failure we condense the source, and this will no longer be possible once the source's length drops below k).

To run this scheme, we will need to use "fresh" random bits for each instantiation of the block extraction scheme. We use l to denote the number of iterations, and assume that the construction process is given l uniformly chosen elements in $\{0,1\}^u$ for running block extraction schemes. The exact recursive construction is given in the following definition.

Definition 13 *For $0 \leq i \leq l$, (where l will be fixed later), we recursively define a sequence of functions, $B_i : \{0,1\}^n \times (\{0,1\}^u)^l \to \{0,1\}^{\frac{n}{r^i}}$, in the following manner:*

- $B_0(x; y_1 \cdots y_l) = x$.
- $B_i(x; y_1, \cdots, y_l) = B(B_{i-1}(x; y_1, \cdots, y_l), y_i)$, *for $i > 0$. (We chose to be a bit informal and ignore the fact that different blocks have different lengths and therefore different u's. Certainly the first block is the longest and therefore the most costly).*

It is easy to see that B_i does not depend on $y_{i+1}, \cdots, y_l$, and that all these computations can be done in polynomial time.

Following the intuition in the introduction, we want to argue that there exists a small l and an $1 \leq i \leq l$ such that $(B_i(X;\cdot), B_{i-1}(X;\cdot))$ is a block source. This is essentially the case, yet formally we will have to partition the space $\{0,1\}^n \times (\{0,1\}^u)^l$ into $l+1$ subsets in an analogous way to lemma 9.

Lemma 10 *There exists a partition of $\{0,1\}^n \times (\{0,1\}^u)^l$ into $l+1$ sets: $BLK_1, \cdots, BLK_l$ and BAD with the following property:*

1. $\Pr(BAD) \leq 2l(\rho + 2^{-t})$

2. (B_i, B_{i-1}) *is a* (k', t)*-block source in* BLK_i*, (where* $k' \geq \frac{\gamma(k-2lt)}{r}$*).*

3. $l = O(\log_r(n/k))$

The proof of lemma 10 works by inductively using lemma 9 and will appear in the full version.

5.2. Condensers

A condenser is a generalization of an extractor. Rather than requiring that it outputs a distribution that is close to uniform it is only required to improve the entropy rate.

Definition 14 *A* (k, k', ϵ)*-condenser is a function* $Con : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{n'}$*, such that for every* k*-source* X *of length* n*, the distribution* $Con(X, U_d)$ *is* ϵ*-close to a* k'*-source, and* $k'/n' > k/n$*.*

Note that an extractor is a special case of a condenser, when $n' = k'$*.*

Had we been able to construct a single block source, we could have used the block source extractor of corollary 3 to get an extractor. Having several candidates, we can run block source extractors on all of them, (using the same seed) and obtain a condenser.

Definition 15 *We define a function* $Con : \{0,1\}^n \times \{0,1\}^{ul+O(\log n)} \to \{0,1\}^{n'}$*. Given inputs* $x \in \{0,1\}^n$ *and* $y \in \{0,1\}^{ul+O(\log n)}$*,* Con *interprets its second argument as* l *strings* $y_1, \cdots, y_l \in \{0,1\}^u$ *and an additional string* s *of length* $O(\log n)$*. For* $0 \leq i \leq l$ *it computes* $b_i = B_i(x; y_1, \cdots, y_l)$*, (where* B_i *is taken from definition 13), and* $o_i = BE(b_i, b_{i-1}, s)$*, (where* BE *is the block source extractor of corollary 3 taking* $m = k'$*). The final output is* $(o_1, \cdots, o_l)$*, (which makes* $n' = lk'$*).*

Intuitively, it is always case that one of the o_i's is close to uniform. Formally, it is straightforward to see that The distribution $(o_1, \cdots, o_l)$ is $(\frac{1}{n} + 2l(\rho + 2^{-t}))$-close to a k'-source. Let us compare the entropy rates of this source and the initial source. The new source has min-entropy k' which is approximately k and length approximately $k \cdot \log \frac{n}{k}$, whereas the initial source had length $n = k \cdot \frac{n}{k}$. This means that Con indeed improves the entropy rate and is a (k, k', ρ)-condenser.

Remark 2 *Actually, the distribution* $(o_1, \cdots, o_l)$ *is a source of a special kind called a "somewhere random source" by Ta-Shma in [16]. In [16] it was shown that extracting randomness from such sources is easier using special extractors which are called "somewhere random mergers". At this point we could have used Ta-Shma's "somewhere random mergers", to extract the randomness from* $(o_1, \cdots, o_l)$*. Instead, we use different methods which are more efficient and exploit the fact that* l *is relatively small.*

We will fix ρ and r to get two condensers with specific parameters. For both of them we fix $\rho = \log^{-3} n$. For the first one we choose $r = 2$. This gives that the condenser maintains a constant fraction of the initial randomness.

Lemma 11 *For every* k*, there exists an explicit* $(k, \Omega(k), 1/\log^2 n)$*-condenser* $Con : \{0,1\}^n \times \{0,1\}^{O(\log \frac{n}{k} \log\log n + \log n)} \to \{0,1\}^{O(k \log \frac{n}{k})}$*.*

For the second condenser we choose $r = 2\log n$. This reduces the number of recursive calls and decreases the number of random bits spent.

Lemma 12 *For every* k*, there exists an explicit* $(k, \Omega(\frac{k}{\log n}), 1/\log^2 n)$*-condenser* $Con : \{0,1\}^n \times \{0,1\}^{O(\log n)} \to \{0,1\}^{\frac{k}{2}}$*.*

Again, we encounter the tradeoff between the length of the seed and the amount of randomness in the output. The first condenser loses only a constant fraction of the randomness, but spends more then $O(\log n)$ random bits, whereas the second spends only $O(\log n)$ random bits, but is able to maintain only a smaller fraction of the randomness.

6. Constructing extractors

In this section we use the condensers constructed in the previous section to prove the two main theorems, (theorems 1,2).

For theorem 1 we use the condenser of lemma 11 repeatedly (with fresh seeds) to condense the source until we achieve constant entropy rate. (This is guaranteed to happen after no more than $\log^* n$ iterations). For constant entropy rate, Zuckerman's extractor, ([22] see table 2), uses the optimal seed length to extract a constant fraction. This procedure loses some randomness in the iteration process, and results an extractor which extracts a sub-constant fraction, $(2^{-O(\log^* n)})$ of the initial randomness. We then use [19] to increase this fraction to an arbitrary constant.

In the case of theorem 2 we are shooting for the optimal seed length and cannot afford the condenser of lemma 11 or repeated condensing. Instead we use the condenser of lemma 12 interpreting it as a block extraction scheme. Viewed this way the condenser extracts a block B of length $k/2$, therefore the distribution (B, X) forms a block source, since B is too short to "steal" all the randomness from X. (This intuition is formalized in the next lemma). All that is left is to use the block source extractor of corollary 3.

Lemma 13 *Let* Con *be the condenser of lemma 12. If* X *is a* k*-source then the distribution* $(Con(X, U_{O(\log n)}), X)$ *is* $O(1/\log^2 n)$*-close to an* $(\Omega(k/\log n), \log^4 n)$*-block source.*

The proof of lemma 13 will be given in the full version.

Remark 3 *The output length in theorem 2, (and corollary 2) can be improved to $k/\log^{\alpha} n$, $(k/\log^{\alpha} k)$. This is done by choosing $r = \log^{\alpha/2} n$ instead of $r = 2\log n$ in lemma 12. This will make the condenser maintain a larger fraction of the initial randomness while still using the optimal seed length. The proof of theorem 2 will now fail, since we used the fact that the condenser has output smaller than k. However, at this point the source becomes so condensed that standard techniques can be used to construct a block source while loosing only another $\log^{\alpha/2} n$ fraction of the randomness.*

7. Achieving small error

The statement of theorems 1,2 is for constant error ϵ. The analysis provided in this paper gives a slightly better result and allows to replace the requirement that ϵ be a constant with $\epsilon = 1/(logn)^c$ for any constant c. Still, our technique does not give good dependence of the seed length on the error[13]. We get better dependence on ϵ using the error reduction transformation of [12], which transforms an extractor with large, (say constant) error into an extractor with arbitrary small error, while losing only a little bit in the other parameters. More precisely, after undergoing the transformation, a factor of $O(\log m(\log\log m)^{O(1)} + \log(1/\epsilon))$ is added to d, and the fraction extracted decreases by a constant. The latter loss makes no difference from our point of view since we are only able to extract constant fractions. The first loss isn't significant in the case of theorem 1, since the seed size is already larger than the optimal one by a multiplicative $polyloglog(n)$ factor. However, it completely spoils theorem 2 and makes it inferior to theorem 1. Here is theorem 1 rephrased using the error reduction transformation of [12]:

Theorem 6 *(theorem 1 rephrased for non-constant ϵ) For every n, k and $\epsilon > exp(\frac{-n}{(\log^* n)^{O(\log^* n)}})$, there are explicit (k, ϵ)-extractors $Ext : \{0,1\}^n \times \{0,1\}^{O(\log n \cdot (\log\log n)^{O(1)} + \log(1/\epsilon))} \to \{0,1\}^{(1-\alpha)k}$, where $\alpha > 0$ is an arbitrary constant.*

8. Acknowledgment

We thank Russel Impagliazzo and Amnon Ta-Shma for helpful discussions, and particularly for bringing to our attention their insights on error correcting random sources.

[13] Although the block extraction scheme of lemma 2 has the required dependence on the error ρ, there are two problems. The first is that we run the block extraction scheme roughly $\log n$ times, (which makes the error the multiplication of the two factors). The second is that because we use Chebychev's inequality our block extraction scheme works only when $\rho = 1/k^{O(1)}$.

References

[1] M. Blum. Independent unbiased coin flips from a correlated biased source—a finite state Markov chain. *Combinatorica*, 6(2):97–108, 1986. Theory of computing (Singer Island, Fla., 1984).

[2] B. Chor and O. Goldreich. Unbiased bits from sources of weak randomness and probabilistic communication complexity. *SIAM J. Comput.*, 17(2):230–261, Apr. 1988.

[3] R. Impagliazzo. private communication, 1999.

[4] R. Impagliazzo, R. Shaltiel, and A. Wigderson. Near optimal conversion of hardness into pseudo-randomness. In *40th FOCS*. IEEE, 1999.

[5] R. Impagliazzo, R. Shaltiel, and A. Wigderson. Extractors and pseudo-random generators with optimal seed length. In *32nd STOC*, 2000.

[6] J. Justesen. A class of constructive asymptotically good algebraic codes. *IEEE Trans. Info. Theory*, 18:652–656, 1972.

[7] J. Naor and M. Naor. Small-bias probability spaces: Efficient constructions and applications. In *22nd STOC*, pages 213–223, Baltimore, Maryland, 14–16 May 1990.

[8] N. 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.

[9] N. Nisan and A. Ta-Shma. Extracting randomness: A survey and new constructions. *J. Comput. Syst. Sci.*, 58(1):148–173, 1999.

[10] N. Nisan and D. Zuckerman. Randomness is linear in space. *J. Comput. Syst. Sci.*, 52(1):43–52, Feb. 1996.

[11] R. Raz and O. Reingold. On recycling the randomness of the states in space bounded computation. In *31st STOC*, Atlanta, GA, May 1999.

[12] R. Raz, O. Reingold, and S. Vadhan. Error reduction for extractors. In *40th FOCS*. IEEE, 1999.

[13] R. Raz, O. Reingold, and S. Vadhan. Extracting all the randomness and reducing the error in Trevisan's extractors. In *31st STOC*, pages 149–158, Atlanta, GA, 1999.

[14] M. Santha and U. V. Vazirani. Generating quasi-random sequences from semi-random sources. *J. Comput. Syst. Sci.*, 33(1):75–87, Aug. 1986.

[15] A. Srinivasan and D. Zuckerman. Computing with very weak random sources. In *35th FOCS*, pages 264–275, Santa Fe, New Mexico, 20–22 Nov. 1994. IEEE.

[16] A. Ta-Shma. On extracting randomness from weak random sources (extended abstract). In *28th STOC*, pages 276–285, Philadelphia, Pennsylvania, 22–24 May 1996.

[17] L. Trevisan. Construction of extractors using pseudo-random generators. In *31st STOC*, Atlanta, GA, May 1999. See also ECCC TR98-55.

[18] J. von Neumann. Various techniques used in connection with random digits. *Applied Math Series*, 12:36–38, 1951.

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

[20] D. Zuckerman. General weak random sources. In *31st FOCS*, volume II, pages 534–543, St. Louis, Missouri, 22–24 Oct. 1990. IEEE.

[21] D. Zuckerman. Simulating BPP using a general weak random source. *Algorithmica*, 16(4/5):367–391, Oct./Nov. 1996.

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