# Dispersers for Affine Sources with Sub-Polynomial Entropy

Ronen Shaltiel
*Department of computer science*
*University of Haifa*
*Haifa, Israel*
*Email: ronen@cs.haifa.ac.il*

**Abstract**— We construct an explicit disperser for affine sources over $\mathbb{F}_2^n$ with entropy $k = 2^{\log^{0.9} n} = n^{o(1)}$. This is a polynomial time computable function $D : \mathbb{F}_2^n \to \{0,1\}$ such that for every affine space $V$ of $\mathbb{F}_2^n$ that has dimension at least $k$, $D(V) = \{0,1\}$. This improves the best previous construction of Ben-Sasson and Kopparty (STOC 2009) that achieved $k = \Omega(n^{4/5})$.

Our technique follows a high level approach that was developed in Barak, Kindler, Shaltiel, Sudakov and Wigderson (J. ACM 2010) and Barak, Rao, Shaltiel and Wigderson (STOC 2006) in the context of dispersers for two independent general sources. The main steps are:

- Adjust the high level approach to make it suitable for affine sources.
- Implement a "challenge-response game" for affine sources (in the spirit of the two aforementioned papers that introduced such games for two independent general sources).
- In order to implement the game, we construct extractors for affine block-wise sources. For this we use ideas and components by Rao (CCC 2009).
- Combining the three items above, we obtain dispersers for affine sources with entropy larger than $\sqrt{n}$. We use a recursive win-win analysis in the spirit of Reingold, Shaltiel and Wigderson (SICOMP 2006) and Barak, Rao, Shaltiel and Wigderson (STOC 2006) to get affine dispersers with entropy less than $\sqrt{n}$.

***Keywords***-Randomness extractors; Dispersers; Pseudorandomness; Explicit construction;

## 1. Introduction

This paper continues an active and long line of research that attempts to construct explicit dispersers and extractors for various families of "imperfect random sources". These objects have found many applications in diverse fields of computer science and mathematics. The reader is referred to survey articles [12], [16], [20], [21].

**Definition 1.1** (dispersers and extractors)**.** *Let $\mathcal{C}$ be a family of distributions over $\{0,1\}^n$ over some set $\Omega$.*

- *$D : \Omega \to \{0,1\}^m$ is a* disperser *for $\mathcal{C}$ with error $\epsilon$ if for every $X \in \mathcal{C}$, $|D(\mathsf{Supp}(X))| \geq (1-\epsilon)2^m$ (where $\mathsf{Supp}(X)$ denotes the support of $X$). In this paper we will be mostly interested in zero-error dispersers for $m = 1$.*
- *$E : \Omega \to \{0,1\}^m$ is an* extractor *for $\mathcal{C}$ with error $\epsilon$ if for every $X \in \mathcal{C}$, $E(X)$ is $\epsilon$-close to the uniform distribution. (where two distributions $X, Y$ over the same set $\Omega$ are $\epsilon$-close if for every event $A$, $|\Pr[X \in A] - \Pr[Y \in A]| \leq \epsilon$).*

This research was supported by ERC starting grant 279559, BSF grant 2010120 and ISF grants 686/07 and 864/11.

Throughout the paper we use the term "source" and "distribution" interchangeably. Many families of sources are considered in the literature, and the goal of this research area is to design explicit (that is polynomial time computable) extractors and dispersers for interesting families. The reader is referred to the survey article [18] and to the introductions of [9], [17] for an overview of the classes of sources considered in the literature.

*Dispersers and extractors for affine sources:* In this paper we consider affine sources. Let $\mathbb{F}_q$ be a finite field of size $q$, and set $\Omega = \mathbb{F}_q^n$. A dimension $k$ affine space in $\mathbb{F}_q^n$ is a set $V = \left\{ \sum_{1 \leq i \leq k} a_i x_i + x' \right\}$ where $x_1, \ldots, x_k \in \mathbb{F}_q^n$ are linearly independent vectors, $a_1, \ldots, a_k \in \mathbb{F}_q$ are scalars, and $x' \in \mathbb{F}_q^n$ is the "shift vector". An affine source $X$ of dimension $k$ is a distribution that is uniformly distributed over some affine space $V$ of $\mathbb{F}_q^n$ with dimension $k$. Note that the dimension of an affine source $X$ is also given by $H(X)/\log q$ where $H$ is the Shannon entropy function.

Past work on dispersers and extractors for affine sources considers two different parameter regimes: For "large fields" with $q = \text{poly}(n)$, Gabizon and Raz, [8] constructed extractors that extract almost all the entropy from affine sources of dimension $k$ for any $k \geq 1$. Gabizon and Shaltiel [9] extended this result to give zero-error dispersers with large output length for the same parameters. DeVos and Gabizon [7] give a tradeoff between the dimension $k$ and the field size $q$ in which decreasing $q$ requires increasing $k$.

The other parameters regime is that of "small fields", and in the extreme, the field $\mathbb{F}_2$. We focus on this setting in the remainder of the paper and identify $\mathbb{F}_2$ with $\{0,1\}$. Note that in this setup the entropy and dimension of affine sources coincide. Constructing dispersers and extractors for affine sources over $\mathbb{F}_2$ turns out to be a challenging problem. Bourgain [5] constructed extractors for affine sources with entropy $k = \delta n$ for every $\delta > 0$. Subsequently, Yehudayoff [22] and Li [11] constructed extractors for affine sources of entropy $\Omega(n/\sqrt{\log \log n})$. Rao [14] constructed extractors

that can handle much lower entropy ($k = \text{polylog}n$) but only for a restricted sub-family of affine sources called "low-weight sources". Li [10] constructed extractors for two independent affine sources, where each one has entropy slightly larger than $\sqrt{n}$. The best known construction of dispersers for affine sources is by Ben-Sasson and Kopparty [3] and can handle affine sources with entropy $\Omega(n^{4/5})$. We also mention that Ben-Sasson and Zewi [4] showed that extractors for affine sources can be used to construct 2-source extractors, and that Demenkov and Kulikov [6] showed that dispersers for affine source yield certain circuit lower bounds.

*Our results:* Note that all previous constructions of extractors and dispersers for general affine sources, need entropy at least $\sqrt{n}$. In this paper we construct dispersers for affine sources with entropy k=$n^{o(1)}$.

**Theorem 1.2.** *There is a polynomial time computable function* $\mathsf{Disp} : \{0,1\}^n \to \{0,1\}$ *that is a zero-error disperser for affine sources of entropy* $k = 2^{\log^{0.9} n}$.

Theorem 1.2 is stated for extracting a single bit. Some of the aforementioned constructions [5], [11], [22] can extract many bits. We remark that using the technique of Gabizon and Shaltiel [9], any zero-error disperser for affine sources that extracts $2\log n$ bits, can be "pushed" to extract almost all the entropy with zero-error. It seems that our technique can be extended to show that for every $\delta > 0$ there is a constant $c_\delta > 0$ and an explicit disperser for $k = n^\delta$ which extracts $c_\delta \cdot \log\log n$ bits. However, at this point we do not know how to extract $2\log n$ bits.

### *1.1. Technique*

In order to explain our approach, we need the following notion of affine subsources.

**Definition 1.3.** *Let $X$ be an affine source. An affine source $X'$ is an affine subsource of $X$ with deficiency $d$ if there exists a linear function $L$ of rank $d$ and a possible value for $v$ for $L(X)$ such that $X'$ is uniformly distributed over $\{x \in \mathsf{Supp}(X) : L(x) = v\}$.*

Our starting point is the following trivial observation:

**Fact 1.4** (Succeeding on an affine subsource)**.** *Let $X$ be an affine source and let $X'$ be an affine subsource of $X$. If* $\mathsf{Disp}$ *is a disperser for $X'$ then* $\mathsf{Disp}$ *is a disperser for $X$.*

Let $\mathcal{P}$ be some property of affine sources. A paradigm for constructing dispersers for affine sources is to show that:

- Every affine source $X$ with sufficiently large entropy has an affine subsource with property $\mathcal{P}$.
- There are explicit dispersers for affine sources with property $\mathcal{P}$.

Let $X$ be an affine source. We will divide $X \in \{0,1\}^n$ into $t$ blocks $X_1, \ldots, X_t \in \{0,1\}^{n/t}$ and consider properties $\mathcal{P}$ of the form "there exist some block $i$ such that $X_i$ has large entropy" (or more generally that the source is nicely structured in the sense that the entropy distribution between different blocks satisfies some condition). For properties $\mathcal{P}$ as above, we can often show that any affine source $X$ with sufficiently large entropy has an affine subsource $X^{(0)}$ that has property $\mathcal{P}$. Furthermore, it is often possible to construct dispersers (or even extractors) for affine sources that have property $\mathcal{P}$, at least assuming that *we know the index* $i$. This does not seem helpful as we get only one sample from the source and cannot hope to find $i$ by looking at one sample.

We follow an approach of [1] and will try to design a procedure $\mathsf{FindBlock}(x)$ with the property that every affine source $X^{(0)}$ that has property $\mathcal{P}$, has a small deficiency affine subsource $X'$ on which $\mathsf{FindBlock}(X')$ identifies index $i$ correctly, with high probability (that is $\Pr[\mathsf{FindBlock}(X') = i] = 1 - o(1)$). Properties $\mathcal{P}$ that we will consider are preserved when conditioning into small deficiency affine subsources, and so $X'$ also has property $\mathcal{P}$ with respect to the *same* index $i$.

Putting things together, let $\mathsf{Disp}(x,i)$ be a disperser for affine sources with property $\mathcal{P}$ (that needs to know the good index $i$). We have that $\mathsf{FindBlock}(X')$ correctly identifies the good index $i$ with high probability, and so after computing $i \leftarrow \mathsf{FindBlock}(x)$ we can apply supply $i$ to $\mathsf{Disp}$ and compute $\mathsf{Disp}(x,i)$. We obtain a disperser for $X'$, which by Fact 1.4 implies is also a disperser for the initial general affine source $X$.

In order to design procedure $\mathsf{FindBlock}$, we follow the high level approach of [1], [2] and design a "challenge-response game" for affine sources. On the one hand, we have an advantage over [1], [2] as affine sources are easier to work with than general sources: We can work with Shannon entropy (rather than so called "min-entropy") and therefore have a well behaved notion of conditional entropy. On the other hand, we have only one source (rather than two), and we use a very restrictive notion of subsources (as we only allow affine subsources). This means that we need to preserve affinity whenever the analysis wants to show the existence of a good subsource. This is in contrast to [1], [2] that work with general sources (that allow distributions over general subsets rather than affine subspaces).[1]

We believe that in addition to the technical contribution of constructing affine dispersers, this paper makes a conceptual contribution in that it demonstrates the usability of the approach of [1], [2] in other settings. Furthermore,

[1]We remark that in an early version of [1] there was a construction of a disperser for affine sources with entropy $\delta n$ for $\delta > 0$. While that construction is the inspiration for this one, the approach used there is very different form the one used here. More specifically, the idea there is to try to reduce the case of affine sources to that of multiple independent sources. The analysis used is very delicate, because it needs to consider general subsources rather than only affine ones. We also remark that the approach of that paper does not make sense for $k < \sqrt{n}$, and even for $k > \sqrt{n}$ it requires very strong components for general sources (that we do not know to construct for $k = o(n)$).

our construction and its analysis are simpler, and require significantly less details and components than that of [1], [2]. It is our view that presenting the approach in the setting of affine sources, makes it easier to grasp. We hope that this will allow subsequent research to apply the method described above for other explicit construction problems.

## 2. PRELIMINARIES

We use small letters as much as possible as we reserve capital letters for random variables. Thus for example, for a matrix $a$ that is defined as a function of $x$, we use $A$ to denote the random variable $a(X)$ for a random variable $X$ that is clear from the context. For a random variable $X$ and an event $E$ with positive probability we use $(X|E)$ to denote the distribution of random variable $X$ when the probability space is conditioned on $E$. The entropy rate of a source $X$ over $\{0,1\}^n$ is $H(X)/n$. We use $[n]$ to denote $\{1,\ldots,n\}$. For $x \in \{0,1\}^n$ and $s \subseteq [n]$ we define $x_s$ to be the $|s|$ bit string given by restricting $x$ to the indices in $s$. We say that a matrix is $r \times c$ if it has $r$ rows and $c$ columns, and use $\{0,1\}^{r\times c}$ to denote the space of such matrices. Thus, $\{0,1\}^{r\times c}$, $\{0,1\}^{c\times r}$ and $\{0,1\}^{r\cdot c}$ denote different things. For a matrix $x$, $x_i$ denotes the $i$'th row of $x$, and for $p \leq c$, $\mathsf{slice}_p(x)$ denotes the $r \times p$ matrix obtained by keeping only the first $p$ columns of $x$.

- *Somewhere-random sources:* A distribution $X$ is $r \times c$, *$\eta$-somewhere random* if $X$ is over $\{0,1\}^{r\times c}$ and there exists $i \in [r]$ such that $X_i$ is $\eta$-close to uniform. (This is equivalent to saying that $X$ is $\eta$-close to a 0-somewhere random distribution). We omit $\eta$ if it is zero, and omit the term "$r \times c$" if it is clear from the context. An affine somewhere random source is a somewhere-random source which is affine (when interpreted as a distribution over $\{0,1\}^{r\cdot c}$).
- *Somewhere-random extractors:* A function $E : \{0,1\}^n \to \{0,1\}^{r\times c}$ is a somewhere-extractor for a family $\mathcal{C}$ with error $\eta$, if for every distribution $X \in \mathcal{C}$, $E(X)$ is $\eta$-somewhere random.
- *Block-wise sources:* A distribution $X$ over $\{0,1\}^{b\times n}$ is an affine block-wise source with entropy threshold $k$ if it is an affine source (when interpreted as a distribution over $\{0,1\}^{b\cdot n}$) and for every $i \in [b]$, $H(X_i|X_1,\ldots,X_{i-1}) \geq k$.
- *Linear seeded strong extractors:* A linear seeded strong $(k,\epsilon)$-extractor is a function $E : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ such that for every $y \in \{0,1\}^d$, $E(\cdot,y)$ is linear, and furthermore for every distribution $X$ that is uniform over a subset of size $2^k$, $(E(X,Y),Y)$ is $\epsilon$-close to uniform (where $Y$ is independent for $X$ and is uniform over $\{0,1\}^d$).

*Useful properties of affine sources:* Throughout the paper we make constant use of the following:

- If $X$ is an affine source with $H(X) \geq k$ and $E$ is a linear seeded strong $(k,\epsilon)$-extractor then for a $(1-2\epsilon)$-fraction of $y \in \{0,1\}^d$, $E(X,y)$ is *completely* uniform.
- If $X$ is an affine source with $H(X) \geq k$ and $X'$ is a deficiency $d$ affine subsource of $X$ then $H(X') \geq k-d$. If $X$ over $\{0,1\}^{2\times n}$ is an affine block-wise source with entropy threshold $k$ and $X'$ is a deficiency $d$ affine subsource of $X$ then $X'$ is an affine block-wise source with entropy threshold $k-d$. We also use the fact that for every possible value $v$ of $X_1$, $H(X_2|X_1 = v) \geq k$.

*Organization of the paper:* Due to space limitations this extended abstract does not contain a proof of Theorem 1.2. The proof is provided in the full version. Instead, we give a full proof of a weaker result in which the entropy threshold is $k = n^{1-\rho}$ for some constant $\rho > 0$.

In Section 3 we show how to implement a "challenge-response game" for affine sources (assuming we can construct certain somewhere-extractors for affine sources). In Section 4 we show how the challenge-response game can be used to find blocks with large entropy, and to construct dispersers for 2-block affine block-wise sources. We use this to implement a dispersers for affine sources with large entropy threshold. In Section 5 we explain how to construct extractors for affine block-wise sources (the full proof is deferred to the full version). We also show how extractors for affine block-wise sources can be used to construct somewhere extractors for affine sources (that we need in order to implement the challenge-response game). In Section 6 we give a high level overview of the ideas used to get dispersers for low entropy.

## 3. CHALLENGE-RESPONSE GAMES FOR AFFINE SOURCES

Let $X$ be an affine source over $\{0,1\}^n$ with $H(X) \geq \delta n$ for some rate $\delta > 0$ which is not necessarily a constant, and let $s \subseteq [n]$ be a subset of size $n'$. We typically think of $X_s$ as a block of $X$, and would like to distinguish between two cases:

- $H(X_s) = 0$ (the block $X_s$ is "empty").
- $H(X_s) \geq \delta n'$ (the block $X_s$ has entropy rate comparable to the initial source $X$).

Obviously, there does not exist a procedure which receives one sample from $X$ and distinguishes the two cases. In this paper, we show how to implement a test that achieves a task with similar flavor. (We will require somewhere-extractors for affine sources as an ingredient).

We start with some high level intuition. We imagine the following game: Block $X_s$ tries to "convince" the source $X$ that it has high rate. For this purpose, block $X_s$ generates a "challenge matrix" $\mathsf{ch}(X_s)$ where $\mathsf{ch}$ is a somewhere-random extractor for rate slightly smaller than $\delta$ so that $\mathsf{ch}(X_s)$ is (close to) somewhere random if $H(X_s) \geq \delta n'$. The source $X$ generates polynomially many response matrices $r_1(X),\ldots,r_{n^2}(X)$ such that at least one of them is completely uniform. This can be done using a linear seeded

strong extractor. The challenge of $X_s$ is responded if there exists a response matrix that equals the challenge matrix. If the challenge is not responded then $X_s$ "wins" (and we say that the test passes). This serves as a demonstration that $H(X_s) \geq \delta n$. Intuitively, any of the $n^2$ response matrices is very unlikely to hit the challenge matrix which is somewhere random. To make this intuition precise we will furthermore show that the challenge matrix is (close to) somewhere random even conditioned on every specific response matrix. This allows us to show that the test passes with high probability if $H(X_s) \geq \delta n'$.

We would like to show that the test fails with high probability if $H(X_s) = 0$. However, this does not hold and indeed we cannot expect to distinguish the two cases using a single sample. We can however show that the test fails with small positive probability as the random response matrix has positive probability to be equal to the challenge matrix (as the latter is fixed if $H(X_s) = 0$). The crux of the approach is to show that if $X$ is a source with $H(X) \geq \delta n$ and $H(X_s) = 0$, then there exists an affine subsource $X'$ of $X$ with low deficiency (so that $H(X') \approx \delta n$) such that on a random sample from $X'$, the challenge is responded with probability one. The interpretation is that the test does correctly identify that the block is empty on the source $X'$ (which is of the same type as $X$).

As explained in the introduction, when constructing dispersers we can imagine that $X = X'$ if we construct a disperser for $X'$. Thus, the statement above suffices for our purposes and it is helpful to imagine that we do have a test that distinguishes the two cases using one sample.

The construction appears below. We formalize the intuition above in Section 3.1 and demonstrate its usability in Section 4.

**Construction 3.1** (Challenge-Response procedure $\mathsf{CR}(x,s)$)**.**

- parameters: *integers* $n, \ell$, $k_{min} \geq (\log n)^a$ *and* $p \leq p_{max} \leq k_{min}^{\alpha}/\ell$ *for some constants* $a > 1$, $\alpha > 0$ *to be determined later.*
- inputs: $x \in \{0,1\}^n$ *and a "block"* $s \subseteq [n]$. *let* $n' = |s|$ *to denote the length of the block.*
- ingredients:
  - *A linear seeded strong* $(k_{min}, 1/4)$*-extractor* $E : \{0,1\}^n \times \{0,1\}^{O(\log n)} \to \{0,1\}^{\ell \cdot p_{max}}$. *By [19] there are explicit linear seed strong extractors* $E$ *which for every* $k_{min} > (\log n)^a$ *(for some constant* $a > 1$*) use seed length* $2 \cdot \log n$ *and output* $k_{min}^{\alpha}$ *bits for some* $\alpha > 0$. *We fix* $E$ *to be this extractor, and note that we previously required that* $\ell \cdot p_{max} \leq k_{min}^{\alpha}$.[2]
  - *A function* $\mathsf{ch} : \{0,1\}^{n'} \to \{0,1\}^{\ell' \times p_{max}}$ *for some* $\ell' \leq \ell$. *(We use several choices for* $\mathsf{ch}$ *in the final construction and so we think of its as a parameter that is provided to the procedure* $\mathsf{CR}$*).*
- Operation: *of procedure* $\mathsf{CR}_{\mathsf{ch},p}(x,s)$
  - *Compute the "challenge matrix"* $c = \mathsf{ch}(x_s)$ *which is an* $\ell' \times p_{max}$ *matrix. We pad* $c$ *with dummy zero rows so that it as an* $\ell \times p_{max}$ *matrix if necessary.*
  - *For all* $y \in \{0,1\}^{2\log n}$ *compute* $r(y) = E(x,y)$ *interpreted as an* $\ell \times p_{max}$ *"response matrix".*
  - *We say that "the challenge is responded at* length $p$*" if there exists a* $y \in \{0,1\}^{2\log n}$ *such that* $\mathsf{slice}_p(c) = \mathsf{slice}_p(r(y))$*, and in that case the output of the procedure* $\mathsf{CR}_{\mathsf{ch},p}(x,s)$ *is defined to be 'fail' and otherwise the output is defined to be 'pass'.*[3]

Construction 3.1 follows the intuitive description above if $p = p_{max}$ so that the function $\mathsf{slice}_p$ is moot. We think of $p$ as a "confidence" that $\mathsf{CR}$ has when it fails. More formally, note that if $p_1 \leq p_2 \leq p_{max}$ and $\mathsf{CR}_{\mathsf{ch},p_2}(x,s) = `fail'$ then $\mathsf{CR}_{\mathsf{ch},p_1}(x,s) = `fail'$. This parameter plays an important role in the disperser construction (and we explain this role later on). However, at this point, we suggest that the reader ignores parameter $p$ and assume that $\mathsf{CR}$ is always applied with the highest confidence parameter $p = p_{max}$.

### *3.1. Properties of CR*

We now show that $\mathsf{CR}$ fails on empty blocks (on some subsource, as explained above). This is captured in the second item of the lemma below. We will explain the role of the first item later on. We stress that the Lemma below makes no specific requirements on the function $\mathsf{ch}$ (other than being a function of the form specified in Construction 3.1).

**Lemma 3.2** ($\mathsf{CR}$ fails on empty blocks)**.** *Let* $n, \ell, p, p_{max}, k_{min}$ *and* $\mathsf{ch}$ *be as in Construction 3.1.*

1) *For every affine source* $X$ *over* $\{0,1\}^n$ *with* $H(X|X_S) \geq k_{min}$, $\Pr[\mathsf{CR}_{\mathsf{ch},p}(X,s) = `fail'] \geq 2^{-\ell \cdot p}$.

[2]The construction described below will go over all seeds of $E$ and therefore it is crucial that $E$ has seed $O(\log n)$. The fact that the leading constant in the seed length is 2 is not crucial. We remark that in the construction of [19], the constant 2 can be replaced with any constant larger than 1 (yielding better efficiency when going over all seeds). We also remark that if we allow seed length $O(\log n)$ for an arbitrary leading constant, we could also use an alternative construction from [19] which has output length $k^{1-\alpha}$ for some constant $\alpha > 0$.

[3]We choose to highlight $\mathsf{ch}, p$ in the notation above as these parameters will vary in the disperser construction, while the other parameters $n, \ell, p_{max}, k_{min}$ will be the same in all applications of $\mathsf{CR}$. This will simplify the notation later on.

2) *For every affine source $X$ over $\{0,1\}^n$ with $H(X) \geq k_{min}$ and $H(X_s) = 0$ there exists a deficiency $\ell \cdot p$ affine subsource $X'$ of $X$ such that $Pr[\mathsf{CR}_{\mathsf{ch},p}(X', s) = \text{`}fail'] = 1$.*

*Proof:* We first note that the first item follows from the second one as for every possible fixing of $x'$ of $X_s$ we have $Y = (X|X_s = x')$ satisfies $H(Y_s) = 0$ and $H(Y) \geq k_{min}$ and we can apply the second item on $Y$ and conclude that $\mathsf{CR}$ fails with probability one on an affine subsource $Y'$ which has deficiency $\ell \cdot p$ and thus has measure $2^{-\ell \cdot p}$ according to $Y$.

We now prove the second item. As $H(X) \geq k_{min}$ and $E$ is a linear seeded strong extractor with entropy threshold $k_{min}$, there exists a seed $y$ such that $R(y) = E(X, y)$ is uniform. Let $T$ denote the event $\{\mathsf{slice}_p(E(X, y)) = \mathsf{slice}_p(\mathsf{ch}(X_s))\}$. We have that $\mathsf{ch}(X_s)$ is fixed and therefore $T$ has probability $2^{-\ell \cdot p}$. Let $X' = (X|T)$ and note that $X'$ is an affine subsource of $X$ with deficiency $\ell \cdot p$ which satisfies the required property. ■

The next lemma captures the intuition that $\mathsf{CR}$ passes on blocks with high entropy. In the aforementioned intuitive description, we were planning to use $\mathsf{CR}$ with a function $\mathsf{ch}$ which is a somewhere-random extractor. Unfortunately, we will not always have an explicit construction of a somewhere-random extractor with suitable parameters. Therefore, we state a weaker requirement on $\mathsf{ch}$ that still suffices for our purposes.

**Definition 3.3.** *Let $X, Y$ be affine sources such that $Y = f(X)$ for some function $f$. We say that $Y$ is* somewhere random *with* error $\eta$ *and* resiliency $r$ *with respect to $X$ if for every linear function $L$ of rank $\leq r$, and every possible output $v$ of $L(X)$, the distribution $(Y|L(X) = v)$ is $\eta$-somewhere random.*

**Lemma 3.4** ($\mathsf{CR}$ passes on blocks with high entropy)**.** *Let $n, \ell, p, p_{max}, k_{min}$ and $\mathsf{ch}$ be as in Construction 3.1. For every affine source $X$ over $\{0,1\}^n$ such that $\mathsf{ch}(X_s)$ is somewhere random with error $\eta$ and resiliency $\ell \cdot p$ with respect to $X$, $\Pr[\mathsf{CR}_{\mathsf{ch},p}(X, s) = \text{`}pass'] \geq 1 - n^2 \cdot (2^{-p} + \eta)$.*

In the informal discussion above we were planning to use $\mathsf{CR}$ to test whether $H(X_s) \geq k'$ for some threshold $k'$. We now explain that a somewhere-extractor $\mathsf{ch}$ with suitable parameters indeed meets the requirement of Lemma 3.4 and yields the aforementioned test. More precisely, if $H(X_s) \geq k'$ and $\mathsf{ch}$ is a somewhere-extractor for affine sources with entropy $k' - \ell \cdot p$ that has error $\eta$, then $\mathsf{ch}(X_s)$ is somewhere random with error $\eta$ and resiliency $\ell \cdot p$ with respect to $X$ as required in the lemma, and therefore $\mathsf{CR}_{\mathsf{ch},p}(X, s)$ passes with high probability. We will always choose parameters so that $\ell \cdot p = o(k')$ (and note that this dictates having the number of output rows $\ell$ in the output of $\mathsf{ch}$ satisfy $\ell \ll k$), and then, a somewhere-extractor $\mathsf{ch}$ for affine sources with entropy $(1 - o(1)) \cdot k'$ can be used to apply $\mathsf{CR}$ and test whether blocks have entropy at least $k'$.

Loosely speaking, the advantage of the weaker condition in Lemma 3.4 is that rather than requiring that $\mathsf{ch}(Y)$ is somewhere random for all affine sources $Y$, it only requires that $\mathsf{ch}(X_s)$ is somewhere random on small deficiency affine subsources of the source $X$ that we are interested in. This is helpful as we will be interested in sources $X$ that have special properties and it will be easier to construct a function $\mathsf{ch}$ that performs well on such sources.

*Proof:* (of Lemma 3.4) For every $y \in \{0,1\}^{2 \log n}$ of $E$, the function $L(x) = \mathsf{slice}_p(E(x, y))$ is linear and has rank $\leq \ell \cdot p$. By the guarantee on $\mathsf{ch}$, for every possible value $v$ of $L(X)$, $(\mathsf{ch}(X)|L(X) = v)$ is $\eta$-somewhere random. Therefore, $\mathsf{slice}_p(\mathsf{ch}(X))$ is somewhere random after fixing $L(X) = \mathsf{slice}_p(R(y)) = \mathsf{slice}_p(E(X, y))$. It follows that for every $y$ and $v$, $\Pr[\mathsf{slice}_p(C) = \mathsf{slice}_p(R(y))|L(X) = v] \leq 2^{-p} + \eta$. Thus, for every $y$, $\Pr[\mathsf{slice}_p(C) = \mathsf{slice}_p(R(y))] \leq 2^{-p} + \eta$, and the Lemma follows by a union bound over the $n^2$ seeds $y \in \{0,1\}^{2 \log n}$. ■

## 4. Using the challenge-response game to construct affine dispersers

### 4.1. Roadmap

In the previous section we saw how to implement the procedure $\mathsf{CR}(x, s)$ that can (in the sense explained above) distinguish the case that $H(X_s) = 0$ from the case that $H(X_s) = k'$ (where $k'$ is a parameter). To implement $\mathsf{CR}$ we require an explicit construction of a function $\mathsf{ch}$, and as discussed above, a somewhere-extractor for affine sources with entropy slightly smaller than $k'$ will do (as long as the number of output rows $\ell$ is sufficiently smaller than $k'$). Unfortunately, we do not know how to construct such somewhere-extractors for affine sources with low entropy. In our final construction we will have to use of weaker objects, and this creates many complications. We can construct a suitable somewhere-extractor for affine sources with entropy $k' = n^{1-\mu}$ for some constant $\mu > 0$, and this construction is explained in Section 5. This in turn can be used to construct dispersers for affine sources with entropy $k = n^{1-\rho}$ for some constant $\rho > 0$. We will present this construction as a warmup in the remainder of this section. This will allow us to explain the ideas that are used in the final construction in a modular way. In Section 6 we give an overview of the ideas needed to achieve $k < \sqrt{n}$. The final construction and analysis are given in the full version.

### 4.2. Nicely structured sources

We will use procedure $\mathsf{CR}$ to find good blocks in "nicely structured sources" that we define next.

**Definition 4.1** (blocks)**.** *A block $s$ is a subset $s \subseteq [n]$ defined by $s = \{a, \ldots, b\}$. We define $\mathsf{left}(s) = \{1, \ldots, a-1\}$. For $t \in [n]$ we define $s_i^t = \{(i-1) \cdot n/t + 1, \ldots, i \cdot n/t\}$. When $t$ is clear from the context we omit it and use $s_i$ or "block $i$" to refer to $s_i^t$.*

**Definition 4.2** (nicely structured source)**.** *Let $n, k'$ be parameters. An affine source $X$ over $\{0,1\}^n$ is* nicely structured *for* block $s$ *with* entropy $k'$ *if $H(X_{\mathsf{left}(s)}) = 0$ and $H(X_s) \geq k'$. We say that $X$ is* strongly-nicely structured *if in addition $H(X|X_s) \geq k'$.*

In a nicely structured source all blocks $j < i$ are empty and block $i$ contains entropy. In a strongly nicely structured source, the pair $(X_s, X)$ forms a 2-block affine block-wise source with entropy threshold $k$. It is easy to see that:

**Lemma 4.3** (Existence of nicely structured affine subsources)**.** *For every $n, k$ and $2 \leq t < k$, every affine source $X$ over $\{0,1\}^n$ with $H(X) \geq k$ has an affine subsource $X^{(0)}$ that is nicely structured for some block $i^*$ with entropy $k/4t$. If furthermore, $n/t \leq k/2$ then $X^{(0)}$ is also strongly nicely structured for block $i^*$ with entropy $k/4t$.*

*Proof:* For every $i \in [t]$, let $k_i = H(X_{s_i}|X_{s_1}, \ldots, X_{s_{i-1}})$. By the chain rule for Shannon entropy, we have that $\sum_{i\in[t]} k_i \geq k$. Let $i^*$ be the smallest $i$ such that $k_i \geq k/4t$. It follows that $\sum_{j<i^*} k_j \leq k/4$. Let $(v_1, \ldots, v_{i^*-1})$ be some value in the support of $(X_{s_1}, \ldots, X_{s_{i^*-1}})$ and set $X^{(0)} = (X|X_{s_1} = v_1, \ldots, X_{s_{i^*-1}} = v_{i^*-1})$. Note that $X^{(0)}$ is an affine subsource of $X$ with deficiency $k/4$ which is nicely structured for block $i^*$ and entropy $k/4t$. We also have that $H(X^{(0)}) \geq H(X) - k/4$. If $n/t \leq k/2$ then $H(X^{(0)}|X^{(0)}_{s_{i^*}}) \geq k - k/4 - n/t \geq k - k/4 - k/2 \geq k/4$ and $X^{(0)}$ is strongly nicely structured. ∎

Note that the two requirements $t < k$ and $n/t \leq k/2$ (which together imply that $X^{(0)}$ is strongly nicely structured) also imply that $k \geq 2n/t > 2n/k$ which gives $k > \sqrt{2n}$. This is necessary as for $t < k < \sqrt{n}$ there does not necessarily exist an affine subsource of $X$ that is strongly nicely structured, as blocks are of length $n/t > n/k > \sqrt{n} > k$ and so it may be that all blocks except for one are empty.

### *4.3. Finding good blocks in nicely structured sources*

Lemma 4.3 says that every affine source $X$ has an affine subsource that is nicely structured at some block $i^*$. Thus, by Fact 1.4, in order to construct dispersers for general affine sources, it suffices to construct dispersers for nicely structured affine sources. If the entropy is sufficiently large, it even suffices to construct dispersers for strongly nicely structured affine sources.

We now observe that procedure $\mathsf{CR}$ can be used to actually *find* a good block $i^*$ in a nicely structured source. This is achieved by the following procedure.

*Procedure* $\mathsf{FindBlock}(x)$*:* We are given $x \in \{0,1\}^n$. For every $1 \leq i \leq t$, apply $\mathsf{CR}(x, s_i)$ (we will explain the precise choice of parameters below) and let $i'$ be the minimal $i$ such that $\mathsf{CR}(x, s_i)$ passes.

Loosely speaking, the intuition is that every application of $\mathsf{CR}$ on a block $j < i^*$ (that is empty) will fail, while the application on block $i^*$ will pass. More formally, in Lemma 4.4 below, we show that every affine source $X$ with $H(X) \geq k$ has an affine subsource $X'$ which is nicely structured at some block $i^*$ with entropy $\approx k/t$, and that furthermore, $\Pr[\mathsf{FindBlock}(X') = i^*] \geq 1 - o(1)$.

This reduces the task of constructing dispersers for general affine sources to constructing dispersers for nicely structured sources in which we *know* the good block $i^*$ (and we will consider this problem in the next section).

In order to implement this approach we need to supply procedure $\mathsf{CR}$ with a somewhere-random extractor $\mathsf{ch}$. In the lemma below we work out the parameters needed for implementing $\mathsf{FindBlock}$. It turns out that with a sufficiently good $\mathsf{ch}$ we can apply $\mathsf{FindBlock}$ even for sources of poly-logarithmic entropy.

**Lemma 4.4** (Correctness of $\mathsf{FindBlock}$)**.** *There is a constant $\alpha > 0$ such that the following holds for every $k \geq polylog(n)$, $t \leq k^{1/3}$ and $\mathsf{ch} : \{0,1\}^{n/t} \to \{0,1\}^{\ell \times p_{max}}$ that is a somewhere extractor for affine sources with entropy $k/16t$ and error $2^{-p_{max}}$. Set $p = p_{max}$ and assume that $\ell, p_{max} \leq k^\alpha$ and $p_{max} \geq 3 \log n$. For every affine source $X$ over $\{0,1\}^n$ with $H(X) \geq k$, there exists an affine subsource $X'$ that is nicely structured for some block $i^*$ with entropy $k/8t$ and furthermore, $\Pr[\mathsf{FindBlock}(X') = i^*] \geq 1 - 2^{-\Omega(p_{max})}$. Moreover, if $n/t \leq k/2$ then $X'$ is strongly nicely structured for block $i^*$*

*Proof:* By Lemma 4.3 there exists an affine subsource $X^{(0)}$ of $X$ that is nicely structured for some block $i^*$ with entropy $k/4t$ (and strongly nicely structured if $n/t \leq k/2$). We choose the parameters $k_{min} = k/8t$ and $p = p_{max}$ in all applications of $\mathsf{CR}$ and note that the requirements of Construction 3.1 are met. We now iteratively go over all $j < i^*$. For each $j$ we apply Lemma 3.2 (2nd item) with respect to source $X^{(j-1)}$ and block $s_j$ to obtain an affine subsource $X^{(j)}$. To apply the Lemma we need to make sure that $H(X^{(j-1)}_{s_j}) = 0$ which happens for every affine subsource of $X^{(0)}$. We also need to verify that $H(X^{(j-1)}) \geq k_{min}$. This holds initially for $X^{(0)}$ and we will maintain the invariant that $X^{(j)}$ is a deficiency $j \cdot \ell \cdot p_{max} \leq k/8t$ deficiency affine subsource of $X^{(0)}$ which implies that $H(X^{(j-1)}_{s_{i^*}}) \geq H(X^{(0)}_{s_{i^*}}) - j \cdot \ell \cdot p_{max} \geq k/4t - k/8t = k/8t$ and consequently $H(X^{(j-1)}) \geq k/8t = k_{min}$. We therefore can apply Lemma 3.2 and conclude that there exists an affine subsource $X^{(j)}$ of $X^{(j-1)}$ of deficiency $\ell \cdot p_{max}$ such that $\Pr[\mathsf{CR}_{\mathsf{ch},p_{max}}(X^{(j)}, s_j) = `fail'] = 1$.

Let $X' = X^{(i^*-1)}$. $X'$ is a deficiency $t \cdot \ell \cdot p_{max} \leq k/8t$ subsource of $X^{(0)}$, and so we have that $H(X'_{s_{i^*}}) \geq k/4t - k/8t \geq k/8t$ and so it is indeed nicely structured for $i^*$ with entropy $k/8t$. If $X^{(0)}$ is strongly nicely structured, we have that $H(X^{(0)}|X^{(0)}_{s_{i^*}}) \geq k/4t$ and as $X'$ is an affine subsource with deficiency $k/8t$ we have that $H(X'|X'_{s_{i^*}}) \geq k/4t -$

$k/8t = k/8t$. we know that $\mathsf{CR}_{\mathsf{ch},p_{max}}(X', s_j)$ fails with probability one for $j < i^*$. Thus, we only need to show that $\mathsf{CR}_{\mathsf{ch},p_{max}}(X', s_{i^*})$ passes with high probability in order to establish that FindBlock identifies $i^*$ correctly. This follows from Lemma 3.4 as we have set up the parameters of ch so that $\mathsf{ch}(X'_{s_{i^*}})$ is somewhere random with resiliency $\ell \cdot p_{max}$. ■

### 4.4. Dispersers for 2-block affine block-wise sources

By the previous discussion we can restrict our attention to affine sources $X$ that are nicely structured for a block $i^*$ that we know. Assume that we somehow achieve that the input source $X$ is in fact *strongly* nicely structured. Note for example, that by Lemma 4.4 above this holds if say, $k \geq n^{3/4}$, $t = n^{1/3}$. Jumping ahead we mention that in our final construction, we will be able to arrange things so that we can assume w.l.o.g. that the source $X$ we work with is strongly nicely structured even for low $k = n^{o(1)}$, and will be able to apply the techniques below for low $k$.

Let $s = s_{i^*}$ and note that by definition $(X_s, X)$ forms a 2-block affine block-wise source with entropy threshold $k_{min}$ for some parameter $k_{min}$. Thus, we are left with the task of designing dispersers for 2-block affine block-wise sources with entropy threshold $k_{min}$.

It turns out that procedure CR is versatile beyond the motivation explained in the previous sections. In fact, setting $\mathsf{Disp}_s(x) = \mathsf{CR}(x, s)$ produces a disperser in case $(X_s, X)$ is a 2-block affine block-wise source (by simply interpreting pass/fail as zero/one).

We summarize the precise parameters in Lemma 4.5 below. We stress that when the entropy threshold of the block-wise source is $k_{min} = k/8t$ (as is the case after applying FindBlock) the parameters with which we run CR below are identical to those chosen in FindBlock and so if we have a somewhere-extractor ch to apply in FindBlock we can reuse it.

**Lemma 4.5** (disperser for 2-block affine block-wise sources)**.** *Let $s \subseteq [n]$ be a subset of size $n'$. Let $X$ be an affine source over $\{0,1\}^n$ so that $(X_s, X)$ forms a 2-block affine block-wise source with entropy threshold $k_{min}$. Let $\mathsf{ch} : \{0,1\}^{n'} \to \{0,1\}^{\ell \cdot p_{max}}$ be a somewhere-random extractor for affine sources with entropy $k_{min} - \ell \cdot p_{max}$ for $\ell \cdot p_{max} \leq k_{min}^{\alpha}$ for some constant $\alpha > 0$ and assume that ch has error $\leq 1/n^3$. For every $3 \log n \leq p \leq p_{max}$ and every $v \in \{`pass', `fail'\}$, $\Pr[\mathsf{CR}_{\mathsf{ch},p}(X, s) = v] \geq 2^{-\ell \cdot p} > 0$.*

*Proof:* Let us analyze the output of $\mathsf{CR}_{\mathsf{ch},p}(X, X_s)$. We have that block $X_s$ has entropy, and this is exactly the setup in which CR is supposed to pass. Consequently, by Lemma 3.4, it follows that $\Pr[\mathsf{CR}_{\mathsf{ch},p}(X, s) = `pass'] \geq 1 - n^2 \cdot (2^{-p} + \eta)$ which is at least $1/2$. We now observe that by Lemma 3.2 (first item), the fact that $H(X|X_s) \geq k_{min}$ implies that $\Pr[\mathsf{CR}_{\mathsf{ch},p}(X, s) = `fail'] \geq 2^{-\ell \cdot p} > 0$. ■

### 4.5. Putting things together: a basic disperser

Putting things together, we have the following construction of a "basic disperser" $\mathsf{BasicD} : \{0,1\}^n \to \{0,1\}$ for affine sources which are strongly nicely structured at some unknown block $i^*$.

*Basic Disperser:* Given input $x \in \{0,1\}^n$

- For every $i \in [t]$ compute $\mathsf{CR}_{\mathsf{ch},p_{max}}(x, s_i)$.
- Apply FindBlock: Namely, let $i'$ denote the smallest $i$ such that the CR passes.
- $\mathsf{BasicD}(x) = \mathsf{CR}_{\mathsf{ch},p}(x, s_{i'})$ where $p \leq p_{max}$ is a parameter that we specify later.

There is a subtlety in making this approach go through. If we use $p = p_{max}$ in the application of CR in the third item, then the two applications of CR on block $X_{s_{i'}}$ are identical. In order for BasicD to output 'fail' we need that the first application outputs 'pass' and the second one outputs 'fail', but this is obviously impossible if the applications are identical.

We overcome this problem by choosing a small confidence parameter $p < p_{max}$. By Lemma 4.4, the probability that the second item fails to identify the good index $i^*$ is at most $2^{-\Omega(p_{max})}$. On the other hand, the probability that the third item outputs a given $v \in \{`pass', `fail'\}$ is at least $2^{-\ell p}$. Thus, taking $p = o(p_{max}/\ell)$ we have that with positive (although small) probability, the block $i^*$ is selected, and the procedure CR fails when applied with confidence level $p$ (and this holds as long as $p \geq 3 \log n$).

*Warmup: A disperser for affine sources of entropy $k = n^{1-\rho}$ for some $\rho > 0$:* For sufficiently large $k$, by Lemma 4.4 every affine source with $H(X) \geq k$ has an affine subsource that is strongly nicely structured, and on which BasicD outputs both pass and fail with positive probability. By Fact 1.4 the procedure BasicD yields a disperser for general affine sources (assuming we can construct a somewhere-extractor ch with suitable parameters). In the next section we construct a somewhere-extractor for affine sources with entropy $k^{1-\mu}$ for some constant $\mu > 0$. Plugging this somewhere-extractor into our machinery we obtain a disperser for affine sources with entropy $k \geq n^{1-\rho}$ for some $\rho > 0$.[4]

## 5. Somewhere extractors via extractors for affine block-wise sources

We need to construct somewhere-extractors for affine sources in order to apply procedure CR. We will construct somewhere-extractors (as well as weaker objects that still suffice for applying CR) using extractors for affine block-wise sources as a building block.

[4]We do not attempt to optimize $\rho$ as this construction is only a warmup for our main construction which achieves $k = n^{o(1)}$. It is clear that the approach we used cannot get $\rho > 1/3$. It seems plausible that one can obtain $\rho \approx 1/4$, but this will require a careful optimization of the constants in some of the components that we use.

### 5.1. An extractor for $O(\frac{\log n}{\log k})$-block affine block-wise source

**Theorem 5.1** (extractor for affine block-wise sources)**.** *There exists a constant $a > 1$ such that for every $n$ and $k > (\log n)^a$ there is a polynomial time computable function $\mathsf{BE} : \{0,1\}^{b \times n} \to \{0,1\}^{k^{\Omega(1)}}$ that is an extractor for $b = O(\frac{\log n}{\log k})$-block affine block-wise sources with entropy threshold $k$, and error $2^{-k^{\Omega(1)}}$.*

The full proof of Theorem 5.1 appears in the full version. We now provide a brief sketch. Our proof relies on ideas developed in [13], [14] and an extractor construction of [14] that works for affine sources which are $k^{\Omega(1)} \times k$ somewhere random. We are given a $b$-block affine block-wise source $X_1, \ldots, X_b$ for $b = O(\frac{\log n}{\log k})$. Our first step is to transform $X_1$ into a somewhere random source with $v = n^{O(1)}$ rows. Each row is obtained by $E(X_1, y)$ where $y$ is one of the $v$ seeds of a linear seeded strong $(k, 1/4)$-extractor $E$ with seed length $O(\log n)$ [19]. We divide the $v$ rows into $v/k^{\Omega(1)}$ chunks where each chunk contains $k^{\Omega(1)}$ rows. We apply the extractor of [14] on each chunk, and as one of the chunks is somewhere random we obtain a source (close to) a somewhere random source with $v/k^{\Omega(1)}$ rows $S_1, \ldots, S_{v/k^{\Omega(1)}}$. We cannot apply the extractor of [14] on $S = (S_1, \ldots, S_{v/k^{\Omega(1)}})$ as it is not affine. Instead, for every $1 \leq i \leq v/k^{\Omega(1)}$ we compute $W_i = E(X_2, S_i)$ where $E$ is a linear seeded strong extractor. If we had that $X_1, X_2$ are independent then we would be guaranteed that $W_i$ is (close to) somewhere random. At this point, we would have consumed source $X_1$, and reduced the number of rows in the somewhere random source by a factor of $k^{\Omega(1)}$. Repeating this process, we could consume $b = O(\frac{\log n}{\log k})$ sources and reduce the number of rows from $n^{O(1)}$ to one.

While the sources $X_1, \ldots, X_b$ are not necessarily independent, we can use the fact that $E$ is linear seeded to argue that the analysis above can be extended to the case that $X_1, \ldots, X_b$ form an affine block-wise source. Loosely speaking, this is because in an affine block-wise source $(X_1, X_2)$ with entropy threshold $k$, $X_2$ can be represented as a sum of two sources where one is a function of $X_1$ and the other is independent of $X_1$. Thus, applying a linear seeded extractor $E(X_2, y)$ where $y$ is a function of $X_1$ can be imagined to be operating on the independent part of $X_2$ once the analysis fixes source $X_1$.

### 5.2. Somewhere-extractors for affine sources

When we partition a source $X$ into $t$ blocks, we can hope that the partition induces a block-wise source. In fact, we will be happy even if the partition induces a block-wise source with $b < t$ blocks as we are shooting to construct somewhere-extractors and therefore don't mind trying all $\binom{t}{b} \leq t^b$ possibilities. This motivates the following definition and Lemma.

**Definition 5.2** (hidden block-wise source)**.** *Let $b \leq k' \leq t \leq n$ be parameters. An affine source $X$ over $\{0,1\}^n$ is a $t$-partition $b$-block affine hidden block-wise source with entropy threshold $k'$ if there exist $i_1 < \ldots < i_b \in [t]$ such that $X_{s_{i_1}}, \ldots, X_{s_{i_b}}$ form a $b$-block affine block-wise source with entropy threshold $k'$. We omit the clause "$t$-partition" if $t$ is clear from the context.*

**Lemma 5.3.** *Let $t \leq k' \leq n$ be parameters. Let $n' = n/t$ and $b = O(\frac{\log n'}{\log k'})$ be the number blocks used in Theorem 5.1 for $BE : \{0,1\}^{b \times n'} \to \{0,1\}^{(k')^{\Omega(1)}}$. Consider the function $\mathsf{SR}(x) = (\mathsf{BE}(x_{s_{i_1}}, \ldots, x_{s_{i_b}}))_{i_1 < \ldots < i_b \in [t]}$. If $X$ is an affine source over $\{0,1\}^n$ that is a $b$-block hidden block-wise source with entropy threshold $k'$, then $\mathsf{SR}(X)$ is $t^b \times (k')^{\Omega(1)}$, $(2^{-(k')^{\Omega(1)}})$-somewhere random.*

Note that $\mathsf{SR}$ runs in polynomial time if $t^b = n^{O(1)}$ (which will always hold for our choices of parameters as we will always have that $b = O(\log n / \log t)$). By an argument similar to that of Lemma 4.3, it is not hard to see that any affine source with sufficiently large entropy is an affine hidden block-wise source.

**Lemma 5.4.** *Let $t \leq k \leq n$ be integers such that $k \geq 2bn/t$ for $b = O(\frac{\log n}{\log k})$ from Lemma 5.3. Every affine source $X$ with $H(X) \geq k$ is a $b$-block affine hidden block-wise source with entropy threshold $k' = k/4t$.*

The function $\mathsf{SR}$ is the component that is missing in the construction for affine dispersers for entropy $n^{1-\rho}$ given in Section 4.5. In the corollary below we set up the parameters for that application.

**Corollary 5.5** (Somewhere-extractor for affine sources with large entropy)**.** *For every sufficiently small constant $\alpha > 0$ there is a constant $\mu > 0$ and a polynomial time computable function $\mathsf{ch} : \{0,1\}^n \to \{0,1\}^{n^{\alpha/3} \times n^{\alpha}}$ that is a somewhere-extractor for affine sources with entropy $n^{1-\mu}$, and has error $2^{-n^{\alpha}}$.*

## 6. High level overview of our construction for $k = n^{o(1)}$

We now outline some of the ideas that will allow us to extend our technique to $k = n^{o(1)}$. We need to solve the following two problems:

- The approach we developed can only work for $k > \sqrt{n}$. This is (amongst other things) because affine sources with entropy $k < \sqrt{n}$ are not guaranteed to have affine subsources which are strongly nicely structured.
- While the machinery we developed in Section 3 can potentially work for $k = n^{o(1)}$, we need to construct a somewhere-extractor $\mathsf{ch}$ for affine sources with entropy $n^{o(1)}$ in order to apply $\mathsf{CR}$ in that parameter regime.

To explain our ideas, it is easier to first assume that we already solved the second problem and can apply $\mathsf{CR}$ to distinguish empty blocks from blocks with entropy rate $\Omega(k/t)$. For simplicity of exposition, let us also cheat

and assume that CR distinguishes empty blocks from high entropy blocks on the original source given to it (without having to consider subsources). We have already seen that informal arguments presented this way can be formalized by carefully considering affine subsources.

*Using a win-win analysis to search for strongly nicely structured sources:* We will solve the first problem using a win-win analysis introduced in [15] (a similar high level approach was also used in [2]). We pick $t = k^{\beta}$ where $\beta$ is a very small constant (or even slightly sub-constant). We show that any affine source over $\{0,1\}^n$ with entropy $k$, has a subsource $X'$ that is either (i) strongly nicely structured with entropy $\Omega(k/t)$, or (ii) nicely structured with entropy threshold $\Omega(k)$. In the discussion below, we will once again ignore subsources and assume that the subsource is the original source. We already constructed dispersers for case (i). In case (ii), for the good block $i^*$, affine source $X_{s_{i^*}}$ is of length $n/t$ and has entropy $\Omega(k)$. This means that $X_{s_{i^*}}$ has entropy rate that is $\Omega(t)$ times the rate of the original source. As $X$ is nicely structured, we can use FindBlock to find block $i^*$ and we now hold a source with better entropy guarantee than the one we started with. We continue this analysis recursively, and at each step either we obtain a strongly nicely structured source, or we multiply the rate by $\Omega(t)$. If this process continues for $O(\log n/\log t)$ steps, then we obtain an affine source of length $n'$ with entropy $k' = (n')^{1-o(1)}$ for which we already constructed dispersers.

*Distinguishing the two cases:* A subtlety that we ignored in the explanation above is that we need to know which of the two cases happened in order to decide whether to stop and produce an output on $X_{s_{i^*}}$ (as in Section 4.4) or continue recursively on block $X_{s_{i^*}}$. We plan to use CR to distinguish case (i) from case (ii). We apply $\mathsf{CR}(X, s_{i^*})$ with a lower confidence parameter $p$ than any of the confidence parameters that we previously used. We need CR to distinguish the case that $\Omega(k/t) \leq H(X_{s_{i^*}}) \ll k$ from the case where $H(X_{s_{i^*}}) \geq \Omega(k)$. Note that in both cases, the block $X_{s_i^*}$ has relatively high entropy and so CR passes with high probability in any of the two cases. Thus, it is very likely that we decide to continue (which is what we want in the second case). We use an argument that is somewhat similar to that used in Section 4.4 in order to argue that if we hold a strongly nicely structured source, then for any Boolean value $v$, there is positive (although small) probability that we stop at this point and output $v$. Thus, we indeed handle both cases correctly.

*Computing challenges recursively:* In the outline above we assumed that we already constructed a function ch that is a somewhere extractor for affine sources of entropy $n^{o(1)}$ with which we can implement CR. We will construct this function ch using once again, a win-win analysis. We remark that we would stop here and the construction and analysis would be much simpler, if we could construct somewhere extractors for affine sources of low entropy.

We show that every affine source $X$ with entropy $k$, has an affine subsource that is either, (i) a $b$-block affine hidden block-wise source with entropy $\Omega(k/t)$, or, (ii) a nicely structured source with entropy threshold $\Omega(k/b) \gg k/t$. In the first case, we can produce a somewhere random matrix by considering all $t^b$ candidate block-wise sources and applying our extractor BE (in a similar way to what is done in Lemma 5.3). In the second case, we can identify the good block using CR and apply ch recursively on a block with significantly higher entropy rate.

This argument may seem circular at first, as we use CR to implement ch and vice-versa. It is important to notice that we apply CR for testing blocks for higher entropy rate than the one we want ch to handle (and this enables us to use recursion). Unlike the previous argument, we do not need to distinguish the two cases (which is crucial as this would give a circular argument if we tried to distinguish the two cases). More precisely, as we are allowed to output a somewhere random source, we simply append the rows of the matrix obtained by the block $s_{i^*}$ that we are considering and the rows of the matrix associated with its chosen sub-block. We know that one of these two matrices is somewhere random and therefore we obtain a somewhere random matrix.

*Extending the argument to the real setup with subsources:* There are subtleties in implementing this recursion when we extend the argument to the true scenario and need to handle subsources. The problem is that we obtain a function ch that is only guaranteed to output a somewhere random distribution on a subsource of the original source that we work with. While this is always good enough when constructing dispersers, it is problematic when constructing extractors (and we want ch to be a somewhere extractor).

To handle this problem, we carefully choose properties of ch that we can maintain recursively, and argue that these properties suffice for applying CR as we go along. We need to be careful in the order in which the analysis goes down to subsources (and this leads to the resiliency property of ch with which Lemma 3.4 is stated).

A little bit more precisely, when given a source $X$ we can first consider an affine subsource $X^{(0)}$ in which the previously described recursion tree, has a node that is strongly nicely structured, and a descendent that is a hidden block-wise source. We can then consider an affine subsource $X'$ of $X^{(0)}$ on which CR fails on all blocks to the left of the blocks that we visit on the way, so that we have a chance to reach the two interesting nodes. This is not a problem, as Lemma 3.2 does not require anything from ch. We now want to apply Lemma 3.4 on the path leading to the two interesting nodes, so that we end up finding the two nodes. The resiliency property stated in Lemma 3.4 can be seen to take care of this problem if we let the resiliency parameter grow with the length of the path.

## 7. Conclusion

We give an explicit construction of dispersers for affine sources with entropy $k = 2^{\log^{0.9} n} = n^{o(1)}$. Interesting open problems are:

- Improve the entropy threshold $k$ to say $\text{polylog}(n)$.
- Increase the number of extracted bits. We remark that by the technique of Gabizon and Shaltiel [9] improving the number of extracted bits to $m = 2\log n$ will automatically give an improvement to $m = \Omega(k)$.
- Our construction would be significantly simpler if one can explicitly construct a somewhere extractor for affine sources with a small number of output rows. More specifically, to get a final result for entropy $k$ we will require a somewhere extractor for entropy $\approx k$ that has a number of output rows which is significantly smaller than $k$, (say $k^{o(1)}$).
- Finally, our methods yield dispersers rather than extractors. It is open to explicitly construct extractors for affine sources with entropy $k = n^{o(1)}$.


## Acknowledgment

I thank Ariel Gabizon and Raghu Meka for inspiring discussions. I am grateful to anonymous referees for many detailed comments and suggestions.


## References


[1] B. Barak, G. Kindler, R. Shaltiel, B. Sudakov, and A. Wigderson, "Simulating independence: New constructions of condensers, ramsey graphs, dispersers, and extractors," *J. ACM*, vol. 57, no. 4, 2010.

[2] B. Barak, A. Rao, R. Shaltiel, and A. Wigderson, "2-source dispersers for sub-polynomial entropy and ramsey graphs beating the frankl-wilson construction," in *STOC*, 2006, pp. 671–680.

[3] E. Ben-Sasson and S. Kopparty, "Affine dispersers from subspace polynomials," in *STOC*, 2009, pp. 65–74.

[4] E. Ben-Sasson and N. Zewi, "From affine to two-source extractors via approximate duality," in *STOC*, 2011.

[5] J. Bourgain, "On the construction of affine extractors," *Geometric And Functional Analysis*, vol. 17, no. 1, pp. 33–57, 2007.

[6] E. Demenkov and A. Kulikov, "An elementary proof of $3n - o(n)$ lower bound on the circuit complexity of affine dispersers," Electronic Colloquium on Computational Complexity, Tech. Rep., 2011.

[7] M. DeVos and A. Gabizon, "Simple affine extractors using dimension expansion," in *IEEE Conference on Computational Complexity*, 2010, pp. 50–57.

[8] A. Gabizon and R. Raz, "Deterministic extractors for affine sources over large fields," *Combinatorica*, vol. 28, no. 4, pp. 415–440, 2008.

[9] A. Gabizon and R. Shaltiel, "Increasing the output length of zero-error dispersers," in *APPROX-RANDOM*, 2008, pp. 430–443.

[10] X. Li, "Improved constructions of three source extractors," in *IEEE Conference on Computational Complexity*, 2011.

[11] ——, "A new approach to affine extractors and dispersers," in *IEEE Conference on Computational Complexity*, 2011.

[12] N. Nisan, "Extracting randomness: How and why a survey," in *IEEE Conference on Computational Complexity*, 1996, pp. 44–58.

[13] A. Rao, "Extractors for a constant number of polynomially small min-entropy independent sources," *SIAM J. Comput.*, vol. 39, no. 1, pp. 168–194, 2009.

[14] ——, "Extractors for low-weight affine sources," in *IEEE Conference on Computational Complexity*, 2009, pp. 95–101.

[15] O. Reingold, R. Shaltiel, and A. Wigderson, "Extracting randomness via repeated condensing," *SIAM J. Comput.*, vol. 35, no. 5, pp. 1185–1209, 2006.

[16] R. Shaltiel, "Recent developments in explicit constructions of extractors," *Bulletin of the EATCS*, vol. 77, pp. 67–95, 2002.

[17] ——, "How to get more mileage from randomness extractors," *Random Struct. Algorithms*, vol. 33, no. 2, pp. 157–186, 2008.

[18] ——, "An introduction to randomness extractors," in *ICALP (2)*, 2011, pp. 21–41.

[19] R. Shaltiel and C. Umans, "Simple extractors for all min-entropies and a new pseudorandom generator," *J. ACM*, vol. 52, no. 2, pp. 172–216, 2005.

[20] S. Vadhan, "Randomness extractors and their many guises," in *FOCS*, 2002, pp. 9–12.

[21] S. P. Vadhan, "The unified theory of pseudorandomness," *SIGACT News*, vol. 38, no. 3, pp. 39–54, 2007.

[22] A. Yehudayoff, "Affine extractors over prime fields," *Manuscript*, 2010.