Electronic Colloquium on Computational Complexity, Report No. 64 (2010)

A New Approach to Affine Extractors and Dispersers

Xin Li *
University of Texas at Austin
lixints@cs.utexas.edu

Abstract

We study the problem of constructing affine extractors over GF(2). Previously the only known construction that can handle sources with arbitrarily linear entropy is due to Bourgain (and a slight modification by Yehudayoff), which relies heavily on the technique of Van der Corput differencing and a careful choice of a polynomial.

In this paper we give a new and arguably simpler construction of affine extractors for linear entropy sources that outputs a constant fraction of the entropy with exponentially small error. This matches the previous best result of Bourgain. The extractor can be pushed to handle affine sources with entropy $n/\sqrt{\log n \log n}$. This slightly improves Bourgain's result and matches the recent result of Yehudayoff. We also give a zero-error disperser for affine sources with entropy $n/\sqrt{\log n}$ that outputs $n^{\Omega(1)}$ bits. This improves previous constructions of affine dispersers that output more than 1 bit.

In contrast to Bourgain's construction, our construction mainly uses extractor machinery and basic properties of polynomials.

*Supported in part by NSF Grant CCF-0634811 and THECB ARP Grant 003658-0113-2007. Part of this work was done while the author was doing an internship at Microsoft Research New England.

ISSN 1433-8092

1 Introduction

Randomness is a very useful resource in computer science. It has been successfully used in many areas such as algorithm design, distributed computing and cryptography. These uses of randomness generally either lead to simpler and more efficient solutions than can be done deterministically, or give solutions where deterministic ones are simply impossible. However, most of these applications typically require the random bits to be uniformly distributed, while it is unclear how to obtain such high quality random bits in the real world. The only thing we can say for sure is that there are phenomena in the natural world that exhibit some kind of randomness, but the probability distribution could vary in arbitrary ways.

The broad area of *randomness extraction* deals with the problem of extracting nearly uniform random bits from some probability distribution. A probability distribution with a certain amount of entropy is called a weak random source. The idea is that if we have an algorithm such that given any weak random source as the input, the output of the algorithm is statistically close to uniform, then we can use any weak random source in any of the applications that typically require uniform random bits. Such an algorithm is called a *randomness extractor*, and the problem of constructing explicit randomness extractors has been studied for many years. We refer the reader to [FS02] for a survey on this subject.

Unfortunately, it is not hard to show that if we don't put any restrictions on the weak random source (except the entropy requirement), then it is impossible to construct a *deterministic* extractor even for weak sources on n bits with entropy $n-1$. Given this negative result, one natural direction is to try to see if we can construct deterministic extractors for special classes of weak sources. Again, it is not hard to show that for any class of weak sources on n bits s.t. the total number of sources is bounded by $2^{\text{poly}(n)}$, there exists a deterministic extractor for this class of sources. Moreover, this extractor can even be computed by a polynomial-sized circuit. The hard part is to give a uniform algorithm that is an extractor for the class of sources.

Over the past years several natural classes of sources have been studied. These include for example samplable sources [TV00], bit-fixing sources [KZ07, GRS04] and small space sources [KRVZ06]. In this paper, we study the class of *affine sources*. Roughly speaking, an affine source with entropy k is a distribution which is uniform over some k-dimensional affine subspace of a vector space $\mathbb{F}^n$.

Definition 1.1. (affine source) Let $\mathbb{F}_q$ be the finite field with q elements. Denote by $\mathbb{F}_q^n$ the n-dimensional vector space over $\mathbb{F}_q$. A distribution X over $\mathbb{F}_q^n$ is an $(n,k)_q$ affine source if there exist linearly independent vectors $a_1, \cdots, a_k \in \mathbb{F}_q^n$ and another vector $b \in \mathbb{F}_q^n$ s.t. X is sampled by choosing $x_1, \cdots, x_k \in \mathbb{F}$ uniformly and independently and computing

$$X = \sum_{i=1}^{k} x_i a_i + b.$$

An affine extractor is a deterministic function such that given any affine source as the input, the output of the function is statistically close to the uniform distribution.

Definition 1.2. (affine extractor) A function $\mathsf{AExt} : \mathbb{F}_q^n \to \mathbb{F}_q^m$ is a deterministic (k,ϵ)-affine extractor if for every $(n,k)_q$ affine source X,

$$|\mathsf{AExt}(X) - U_m| \leq \epsilon.$$

Here U_m is the uniform distribution over $\mathbb{F}_q^m$.

In this paper we focus on the case where $q = 2$. Using the probabilistic method, it is not hard to show that there exists a deterministic affine extractor, as long as $k > 2\log n$ and $m < k - O(1)$. The problem is to given an explicit construction of such a function.

A weaker version of the extractor, called an affine disperser, only requires the output to have a large support size.

Definition 1.3. (affine disperser) A function $\mathsf{ADisp} : \mathbb{F}_q^n \to \mathbb{F}_q^m$ is a deterministic (k, ϵ)-affine disperser if for every $(n, k)_q$ affine source X,

$$|\mathsf{Supp}(\mathsf{ADisp}(X))| \geq (1 - \epsilon)q^m.$$

The function is called a zero-error disperser if $\epsilon = 0$.

In this paper we focus on the case of zero-error affine dispersers.

The study of affine extractors and dispersers started with the work of Gabizon and Raz [GR05], where they constructed explicit extractors for affine sources even with entropy 1. However, their constructions require the field size to be much larger than n, i.e., $q > n^{\Omega(1)}$. As the field size gets smaller, the problem of constructing explicit affine extractors gets harder. Recently DeVos and Gabizon [DG10] constructed explicit extractors for $(n, k)_q$ affine sources when $q = \Omega((n/k)^2)$ and the characteristic of the field $\mathbb{F}_q$ is $\Omega(n/k)$.

In the hardest case where $q = 2$ and the field is $\mathbb{F} = \mathrm{GF}(2)$, it is well known how to construct extractors for affine sources with entropy rate greater than $1/2$. However the problem becomes much harder as the entropy rate drops to $1/2$ and below $1/2$. The best construction of affine extractors in this case is due to Bourgain [Bou07], who gave an extractor for affine sources with arbitrarily linear entropy that can output a constant fraction of the entropy with exponentially small error. Based on Bourgain's techniques, recently Yehudayoff [Yeh10] gave another construction in the same spirit that is slightly simpler to analyze. The construction of [Yeh10] also slightly improves Bourgain's result. Rao [Rao09] constructed extractors for affine sources with entropy as small as polylog(n), as long as the subspace of X has a basis of low-weight vectors.

The construction of [Bou07], and the slight modification of [Yeh10], are thus the only known constructions for general affine sources over GF(2) with arbitrarily linear entropy. However, both of these constructions rely heavily on the technique of Van der Corput differencing. Also, both of these constructions need to choose a polynomial very carefully, so that eventually the Van der Corput differencing would result in an estimate of exponential sums in finite fields. The polynomial chosen thus determines the performance of the extractor. From our point of view the choice of the polynomial is somewhat subtle and a bit unnatural. Thus one may ask the natural question of whether there exist other constructions of affine extractors for arbitrarily linear entropy sources.

In this paper we give a new construction of affine extractors that matches the results of [Bou07] and [Yeh10]. Our construction mainly uses tools from previous constructions of extractors and produces the desired polynomial in a very natural way. This gives new insights into the nature of affine extractors. We also believe that having two different constructions with the best known parameters more than double the chances of achieving even better constructions.

In the case of constructing dispersers for affine sources over GF(2), Barak et al. [BKS+05] gave an affine disperser for sources with arbitrarily linear entropy that outputs a constant number of bits. Ben-Sasson and Kopparty [BSK09] constructed dispersers for affine sources with entropy $\Omega(n^{4/5})$. However, their construction only outputs 1 bit. In this paper, we construct dispersers for slightly sub-linear entropy affine sources that output $n^{\Omega(1)}$ bits.

1.1 Our Results

Our first two results give affine extractors that match the constructions of Bourgain [Bou07] and Yehudayoff [Yeh10]. Specifically, we have

Theorem 1.4. *For every $\delta > 0$ there exists an efficient family of functions $\mathsf{AExt} : \{0,1\}^n \to \{0,1\}^m$ such that $m = \Omega(n)$ and for every affine source X with entropy δn, $|\mathsf{AExt}(X) - U_m| = 2^{-\Omega(n)}$.*

Theorem 1.5. *There exists a constant $c > 1$ and an efficient family of functions $\mathsf{AExt} : \{0,1\}^n \to \{0,1\}^m$ such that $m = n^{\Omega(1)}$ and for every affine source X with entropy $cn/\sqrt{\log \log n}$, $|\mathsf{AExt}(X) - U_m| = 2^{-n^{\Omega(1)}}$.*

Our third result gives a zero error affine disperser. Note that an affine extractor is also an affine disperser. Our construction of affine dispersers improves the results of [BKS+05], [Bou07] and [Yeh10]. Although the entropy that our affine disperser can handle is not as small as that of [BSK09], our construction has the advantage of outputting $n^{\Omega(1)}$ bits, while in [BSK09] the construction only outputs 1 bit.

Theorem 1.6. *There exists a constant $c > 1$ and an efficient family of functions $\mathsf{ADisp} : \{0,1\}^n \to \{0,1\}^m$ such that $m = n^{\Omega(1)}$ and for every affine source X with entropy $cn/\sqrt{\log n}$, $|\mathsf{Supp}(\mathsf{ADisp}(X))| = 2^m$.*

1.2 Overview of Our Constructions and the Analysis

In some sense, our construction is similar in the spirit to the construction of 2-source extractors by Kalai et al. [KLR09]. There the authors constructed an extractor for two independent general weak random sources with linear entropy, based on a computational assumption. The basic idea is that, when a weak random source with linear entropy is divided into some constant number of blocks, it becomes (up to a convex combination and a small error) a somewhere block source. If we know the location of the first good block (a block that contains high entropy but not all the entropy of the source), then it is fairly easy to extract random bits since we already have extractors for a weak source and an independent block source. The problem is that we don't know where the good block is, thus we have to try all the possibilities and we get a constant number of outputs such that one of them is close to uniform. We could now take the xor of these outputs but there is no guarantee that the result is close to uniform since the outputs can be correlated. The authors then modified this procedure by using computational assumptions to make the outputs "independent" of each other in some sense, and thus the xor of the outputs is close to uniform.

In this paper, we would like to do something similar. It is still true that when we divide an affine source with linear entropy into some constant number of blocks, it becomes an affine somewhere block source. However, unlike in [KLR09], we now have two problems. First, we have only one affine source. Second, we do not rely on any computational assumptions. Thus we need some new techniques to deal with these problems.

1.2.1 Extractors for one affine block source

Our starting point is the nice structure that an affine source exhibits under a linear function. Roughly speaking, if X is an affine source and L is a linear function, then there exist two independent affine sources A and B such that $X = A + B$ and $\forall b \in \mathsf{Supp}(B)$, $L(b) = 0$ (see Lemma 2.10).

To construct an extractor for an affine block source, we use special kinds of two-source extractors and seeded extractors: strong linear two-source (or seeded) extractors. A two-source (or seeded) extractor is strong if for most choices of one source (or the seed), the output is close to uniform. It is linear if for any fixing of one source (or the seed), the output is a linear function of the other source. There are known strong linear seeded and two-source extractors. For example, Trevisan's extractor [Tre01] is a strong linear seeded extractor, while the inner product function is a strong linear two-source extractor when the sum of the entropy rates of the two sources is greater than 1.

Given these extractors, our extractor for an affine block source is simple. Assume (X_1, X_2) is an affine block source with each block having entropy rate δ. We first use the condenser based on sum-product theorems to convert X_1 into a somewhere rate-$(1-\delta/2)$ source, which is a matrix that has a constant number of rows such that one of the rows has entropy rate $1-\delta/2$. Next we apply the inner product function to each row and X_2. Although X_1 and X_2 might be correlated, note that X_1 is a linear function of the source (X_1, X_2). Thus the structure of affine sources (Lemma 2.10) and the properties of strong linear two-source extractors guarantee that the output is close to a convex combination of affine somewhere random sources.

Note the affine somewhere random source has very few rows (a constant number), thus we can now use Rao's extractor for such sources [Rao09]. Rao's extractor uses the strong linear seeded extractor, and reduces the number of rows in the affine somewhere random source by a half each time, while keeping it to be a convex combination of affine somewhere random sources. Thus by repeating a constant number of times we get an output R that is close to uniform.

In the real construction, we use the output R as a seed and apply the strong linear seeded extractor again to the source to get another output U that is close to uniform. The purpose of doing this is to make sure that the output is a linear function of the source (when conditioned on the fixings of some random variables), thus we could use the structure result in Lemma 2.10.

Again, the problem is that we don't know where the good block is. Thus we have to try all the possibilities and get a constant number of outputs U_i such that one of them is close to uniform.

Remark 1.7. In this construction, the extractor for an affine somewhere random source in [Rao09] needs to use an extractor for affine sources with entropy rate 1/2 as a black box. In this paper we use the simple constructions in [Bou07]. There are two such constructions in [Bou07]. Roughly speaking, if we divide X into 3 equal blocks X_1, X_2, X_3 or 2 equal blocks X_1, X_2, and take the first $\Omega(n)$ bits of $X_1X_2X_3$ or $(X_1X_2)^3$, then the output is exponentially close to uniform. Both constructions work for affine sources with entropy rate slightly below 1/2.

This is the only place that our construction may rely on Van der Corput differencing and the estimate of exponential sums in finite fields. Moreover, we only rely on them in a black box manner. We note that there are other constructions of affine extractors that achieve entropy rate below 1/2. For example, the authors of [BSK09] showed that $\mathsf{Tr}(x^7)$ is an affine extractor for entropy rate roughly 2/5. However, their construction seems to output just one bit and thus does not meet our needs. On the other hand, since our construction uses such an extractor as a black box, any construction of affine extractors for entropy rate 1/2 in the future can be used in our construction to give new affine extractors for small entropy rate.

1.2.2 Obtain one output

Similar to the construction in [KLR09], we need to find a way to make the outputs somewhat "independent" of each other, so that we can take the xor of them and get a string that is close to

uniform. To do this, we are going to use properties of polynomials over GF(2).

First of all, in the analysis we can fix the first good block X_g (though we don't know which block it is) and fix all random variables produced before this block. By restricting the size of the random variables produced (so that they don't steal much entropy), it is not hard to show that conditioned on all these fixings, with very high probability U_g is still close to uniform (in fact, in this case we can show that U_g is actually perfectly uniform). Thus in particular this means that conditioned on $(U_1, \cdots, U_{g-1})$, U_g is still close to uniform. The problem is that $U_{g+1}, \cdots, U_t$ (t is the number of blocks) could be correlated with U_g. Thus there's no guarantee that the xor of these random variables is close to uniform.

Our key observation is that, when we fix all the random variables produced before X_g and fix (X_g, R_g), U_g is a linear function L of the source X. Lemma 2.10 thus tells us that there exists an affine function L_g and an affine source B_g independent of U_g such that $X = L_g(U_g) + B_g$ (intuitively, B_g is the affine source whose support is $\mathsf{Ker}(L) \cap \mathsf{Supp}(X)$, and L_g is the inverse function of L). Therefore, now conditioned on any fixing of $B_g = b$ the source X is an affine function (degree 1 polynomial) of U_g!

Since the subsequent computations are all functions of X, the random variables $U_{g+1}, \cdots, U_t$ are all functions of U_g. We note that any boolean function on $\{0,1\}^n$ can be expressed as a polynomial over GF(2), though the degree of the polynomial can be very high. On the other hand, if we can ensure that each bit of $U_{g+1}, \cdots, U_t$ is a low degree (say degree d) polynomial of the bits of U_g, then there is something natural to do–instead of just outputting U_i, we output a bit Z_i, which is degree d_i polynomial of the bits of U_i. Now $Z_{g+1}, \cdots, Z_t$ are polynomials of degree $d \cdot d_{g+1}, \cdots, d \cdot d_t$ of the bits of U_g. As long as $d_g > max\{d \cdot d_{g+1}, \cdots, d \cdot d_t\}$, the xor of the Z_i's cannot be a constant. Thus we get one bit that can take both values of $\{0,1\}$. In other words, we get a one-bit disperser! In fact, in the construction we are going to choose $d_i > d_{i+1}$ for all i. Thus as long as each bit of $U_{g+1}, \cdots, U_t$ is a degree d polynomial of the bits of U_g, it suffices to have $d_g > d \cdot d_{g+1}$. Since we don't know which block is the first good block, we take $d_i > d \cdot d_{i+1}$ for all i.

In the construction for linear entropy affine sources, the degree d is a constant. Thus we can take all the d_i's to be constants. The polynomial Z_i we take is simple too: just take d_i bits of U_i (say the first d_i bits) and compute the product. We'll show that each U_i has $\Omega(n)$ bits, thus we can take $\Omega(n)$ different blocks of d_i bits. Therefore instead of just outputting one bit, we can output $\Omega(n)$ bits. The only thing left now is to make sure each bit of $U_{g+1}, \cdots, U_t$ is a low degree polynomial of the bits of U_g.

1.2.3 Extractors that are low-degree polynomials

Let us examine how U_i is computed. First we convert a block X_i of X into a somewhere rate-$(1-\delta/2)$ source Y_i, using the condenser based on sum-product theorems. In this step we need to apply the condenser a constant number of times, while each time the output is a degree 2 polynomial of the inputs. Next we apply the inner product function to each row of Y_i and X to obtain an affine somewhere random source SR_i. Again the output is a degree 2 polynomial of the inputs. We then use the extractor for such a source from [Rao09] to get a random seed R_i. Finally we use R_i and apply a strong linear seeded extractor to X to obtain U_i. In these two steps, the affine extractor for entropy rate 1/2 used is a constant degree polynomial, but it may not be the case for the strong linear seeded extractor.

We note that in the above discussion some of the polynomials are over a finite field $\mathbb{F}_q$. However, in this paper all the finite fields $\mathbb{F}_q$ we use have size $q = 2^s$ for some integer s. Thus by mapping a

string in $\{0,1\}^s$ to an element in $\mathbb{F}_q$ using the standard binary expression, we see that whenever a function is a degree d polynomial over $\mathbb{F}_q$, each bit of the output is also a degree d polynomial (over GF(2)) of the input bits. Therefore all we need now is to make sure in the strong linear seeded extractor, each bit of the output is a constant degree polynomial of the input bits. Trevisan's extractor is a strong linear seeded extractor, however each bit of the output is a degree $\Omega(\log n)$ polynomial of the bits of the inputs. Thus it is not suitable for our application. In this paper we construct a new strong linear seeded extractor, with the property that the output is a constant degree polynomial of the inputs.

The starting point is the well-known leftover hash lemma [ILL89], which roughly says that if R is uniformly distributed over a finite field $\mathbb{F}$ and X is a weak source over $\mathbb{F}$, then the last several bits of $R \cdot X$ (the operation is in $\mathbb{F}$) is a strong extractor. Note this is also a linear seeded extractor and the output is a degree 2-polynomial of the inputs. The only bad thing about this extractor is that it requires the seed to have as many bits as the source, which we cannot afford. Nevertheless, we use this extractor as an ingredient in our construction.

Our actual construction of the strong linear seeded extractor consists of three steps. First, we take a short seed and divide the source into many blocks with each block having the same number of bits as the seed. We apply the extractor from the leftover hash lemma to the seed and every block of the source, and concatenate the outputs. Thus we get a somewhere random source. Next, we take another short seed and apply the strong linear merger from [LRVW03] to the somewhere random source. We then get a short source with linear entropy. Finally, we take a short seed and apply the extractor from the leftover hash lemma to the short source, and we obtain bits that are close to uniform. It is easy to check that the extractor is a linear seeded extractor since in all three steps, conditioned on any fixing of the seed the output is a linear function of the source. It is also strong because in each step the extractor or merger is strong. It can be easily checked that the output is a degree 4 polynomial of the inputs.

Once we use this extractor, it is fairly straight forward to check that each bit of U_i is a constant degree polynomial of the bits of X, and thus the bits of U_g.

Remark 1.8. Actually, even if we use Trevisan's extractor as the strong linear seeded extractor in our construction, we still get an affine disperser for linear entropy sources. To see this, note that in Trevisan's extractor, each bit of the output is a degree $O(\log n)$ polynomial of the bits of the inputs. In the extractor for an affine somewhere random source, we need to repeat Trevisan's extractor for a constant number of times. Thus each bit of U_i is a degree polylog(n) polynomial of the bits of X, i.e., $d = \text{polylog}(n)$. Therefore we can still take $d_i > d \cdot d_{i+1}$ for all i, but we can only output $n/\text{polylog}(n)$ bits. However, to get an affine extractor we cannot afford to use polynomials of degree polylog(n). The reason is explained in the following discussions.

1.2.4 Affine Extractors

Above we discussed the techniques in our construction of affine dispersers. Next we discuss how to get an affine extractor. Our affine extractor is a modification of our construction of affine dispersers.

Recall that in the above we take Z_g to be a degree $d_g > d \cdot d_{g+1}$ polynomial of the bits of U_g (the product of the first d_g bits) and we argue that Z_g xored with $Z_{g+1}, \cdots, Z_t$ cannot be a constant. The key observation here is that this is not only true for $Z_{g+1}, \cdots, Z_t$, but also true for any polynomial of degree at most $d_g - 1$. In other words, let $d' = d_g - 1$ and let $P_{d'}$ stand for the class of all polynomials of degree at most d' of the bits of U_g, then the correlation between Z_g and

$P_{d'}$ is at most $1 - 1/2^{d'}$. Therefore if we take several independent blocks of U_g, each having d_g bits, and take the xor of the products of the bits in each block, the correlation with $P_{d'}$ will decrease exponentially by the xor lemma from [VW08, BKS+09]. Since U_g has $\Omega(n)$ bits we can take $\Omega(n)$ blocks and the correlation will decrease to $2^{-\Omega(n)}$. Thus we get an extractor that outputs one bit with exponentially small error.

To output more bits, we divide the bits of U_g into $\Omega(n)$ blocks, each having d_g bits. We next take the generating matrix of a binary linear asymptotically good code, with the codeword length equaling the number of blocks. For each row of the generating matrix we associate one bit. The bit is computed by taking the xor of the products of the bits in the blocks that are indexed by the 1's in this row. By the properties of the asymptotically good linear code, the xor of any non-empty subset of these bits, will be the xor of the products of the bits from $\Omega(n)$ blocks. Thus it will be $2^{-\Omega(n)}$-close to uniform. In other words, these bits form a $2^{-\Omega(n)}$-biased space. Therefore we can take $\Omega(n)$ bits that are $2^{-\Omega(n)}$-close to uniform.

1.2.5 Constructions for Sub-Linear Entropy

Our constructions of affine extractors and dispersers can be easily extended to handle sources with slightly sub-linear entropy. The main point is that in the constructions for sources with linear entropy, the polynomials we use have constant degrees, while each U_i has $\Omega(n)$ bits. Therefore we can deal with sub-linear entropy sources by using polynomials of higher degrees.

More specifically, for an affine source with entropy rate δ, we need to divide the source into $O(1/\delta)$ blocks to ensure that it is a somewhere block source. We'll show that each bit of U_i is roughly a degree $2^{O(1/\delta)}$ polynomial of the bits of X. Thus if we choose $d_i > d \cdot d_{i+1}$ with $d = 2^{O(1/\delta)}$ for all $i = 1, \cdots, O(1/\delta)$, we get that the maximum degree of the polynomials is roughly $2^{O(1/\delta^2)}$. Note that we choose the polynomial Z_i of U_i by taking the product of d_i different bits. Thus we need to make sure that U_i has more than d_i bits. For the case of dispersers, we can take δ as small as $\Omega(1/\sqrt{\log n})$, which gives a maximum degree of $n^{\Omega(1)}$ and outputs $n^{\Omega(1)}$ bits. For the case of extractors, we can only choose the degree as large as $\log n$ since this is the largest degree for which the xor lemma of [VW08, BKS+09] gives a non-trivial correlation bound. Therefore in this case we can take δ as small as $\Omega(1/\sqrt{\log \log n})$.

Remark 1.9. In our construction of a strong linear seeded extractor, we use the linear merger from [LRVW03]. We can also use the curve merger from [DW08, DKSS09]. The curve merger is also a linear function for any fixing of the seed, but each bit of the output is a degree $r + 1$ polynomial of the input bits, where r is the number of rows in the somewhere random source. On the other hand, the curve merger has shorter seed length.

If we use the curve merger, then for linear entropy affine sources it gives essentially the same result, since the somewhere random sources always have a constant number of rows. For sub-linear entropy sources the entropy it can handle is slightly worse: it gives affine extractors for entropy roughly $n/\sqrt[3]{\log \log n}$ and affine dispersers for entropy roughly $n/\sqrt[3]{\log n}$. In fact, we can use other mergers too. For example, we can first divide the rows of the somewhere random source into several blocks, and apply the curve merger to each block. We can then apply the curve merger or the linear merger to the outputs of the blocks. We can even repeat this procedure several times. This will give us a merger that has degree and seed length between the linear merger and the curve merger. Therefore, our construction actually produces a class of polynomials that are affine extractors for arbitrarily linear (and slightly sub-linear) entropy sources.

1.3 Conclusions and Open Problems

In this paper we give new constructions of affine extractors and dispersers over GF(2) that match (and slightly improve) previous best constructions that output more than 1 bit. There are several natural open problems left.

The first and most obvious open problem is to construct affine extractors and dispersers for sources with smaller entropy. We note that the limit of our affine extractors is due to the fact that the xor lemma of polynomials in [VW08, BKS+09] can only handle polynomials of degree smaller than $\log n$. Thus an improvement of the xor lemma would improve our extractors too. For the disperser case we don't need the xor lemma, and we only need a function Z_i of U_i that doesn't have correlation 1 with the xor of $Z_{i+1}, \cdots, Z_t$. This gives some hope of further improvements. It is also worthwhile to see if we can combine the techniques in this paper with the techniques of [Bou07], [Yeh10] and [BSK09] to give better constructions of affine extractors and dispersers.

Second, it is interesting to compare the techniques in this paper to the techniques of the 2-source extractor construction in [KLR09]. Both constructions first obtain a constant number of outputs such that one of them is close to uniform, then find the right tools to remove the correlations between these outputs, so that the xor of the outputs is close to uniform. In [KLR09], the authors used one way permutations for weak random sources and reconstructive extractors. In this paper, because of the special structure of affine sources, we use strong linear seeded extractors and xor lemmas of polynomials. The results suggest that we may be able to apply these techniques to other classes of sources too.

1.4 Organization of this Paper

The rest of the paper is organized as follows. In Section 2 we review some basic definitions and the relevant background. In Section 3 we describe our construction of a constant degree strong linear seeded extractor. Section 4 describes the extractor for affine somewhere random sources, using as an ingredient our strong linear seeded extractor. In Section 5 we give our main constructions of an affine disperser and an affine extractor for linear entropy sources. Finally in Section 6 we briefly show how we can generalize the constructions to handle slightly sub-linear entropy.

2 Preliminaries

We use common notations such as $\circ$ for concatenation and $[n]$ for $\{1, 2, \cdots, n\}$. All logarithms are to the base 2. We often use capital letters for random variables and corresponding small letters for their instantiations.

2.1 Basic Definitions

Definition 2.1 (statistical distance)**.** Let D and F be two distributions on a set S. Their **statistical distance** is

$$|D - F| \stackrel{def}{=} \max_{T \subseteq S}(|D(T) - F(T)|) = \frac{1}{2}\sum_{s \in S}|D(s) - F(s)|$$

If $|D - F| \leq \epsilon$ we say that D is ϵ-*close* to F.

Definition 2.2. The *min-entropy* of a random variable X is defined as

$$H_\infty(X) = min_{x \in \mathsf{supp}(X)}\{-\log_2 \Pr[X = x]\}.$$

We say X is an (n, k)-source if X is a random variable on $\{0,1\}^n$ and $H_\infty(X) \geq k$. When n is understood from the context we simply say that X is a k-source.

In the case of affine sources, the source is a uniform distribution over an affine subspace. Thus the min-entropy is the same as the standard Shanon entropy, and we simply use $H(X)$ to stand for the entropy of an affine source.

2.2 Somewhere Random Sources, Mergers and Condensers

Definition 2.3 (Somewhere Random sources)**.** A source $X = (X_1, \cdots, X_t)$ is (r, t) *somewhere-random* (SR-source for short) if each X_i takes values in $\{0,1\}^r$ and there is an i such that X_i is uniformly distributed.

Definition 2.4. An elementary somewhere-k-source is a vector of sources $(X_1, \cdots, X_t)$, such that some X_i is a k-source. A somewhere k-source is a convex combination of elementary somewhere-k-sources.

Definition 2.5 (Merger)**.** A function $M : (\{0,1\}^n)^s \times \{0,1\}^d \to \{0,1\}^n$ is called an (m, ϵ)-merger (of (n, s)-somewhere-random sources), if for every (n, s)-somewhere random source $X = (X_1, \cdots, X_s)$, and for R being uniformly distributed over $\{0,1\}^d$, the distribution of $M(X, R)$ is ϵ-close to having min-entropy m. We say that the merger is strong if the average over $r \in \{0,1\}^d$ of the statistical distance between $M(X, r)$ and an (n, m)-source is $\leq \epsilon$.

Definition 2.6. A function $C : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is a $(k \to l, \epsilon)$-condenser if for every k-source X, $C(X, U_d)$ is ϵ-close to some l-source. When convenient, we call C a rate-$(k/n \to l/m, \epsilon)$-condenser.

Definition 2.7. A function $C : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is a $(k \to l, \epsilon)$-somewhere-condenser if for every k-source X, the vector $(C(X, y)_{y \in \{0,1\}^d})$ is ϵ-close to a somewhere-l-source. When convenient, we call C a rate-$(k/n \to l/m, \epsilon)$-somewhere-condenser.

2.3 Strong Linear Seeded Extractors

We need the following definition and property of a specific kind of extractors.

Definition 2.8. A function $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is a *strong seeded extractor* for min-entropy k and error ϵ if for every min-entropy k source X,

$$\Pr_{u \leftarrow_R U_d}[|\mathsf{Ext}(X, u) - U_m| \leq \epsilon] \geq 1 - \epsilon,$$

where U_m is the uniform distribution on m bits. We say that the function is a *linear strong seeded extractor* if the function $\mathsf{Ext}(\cdot, u)$ is a linear function over $\mathsf{GF}(2)$, for every $u \in \{0,1\}^d$.

Proposition 2.9 ([Rao09])**.** *Let* $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *be a linear strong seeded extractor for min-entropy* k *with error* $\epsilon < 1/2$*. Let* X *be any affine source with entropy* k*. Then,*

$$\Pr_{u \leftarrow_R U_d}[|\mathsf{Ext}(X, u) - U_m| = 0] \geq 1 - \epsilon$$

2.4 The Structure of Affine Sources

The following lemma explains the behavior of a linear function acting on an affine source.

Lemma 2.10. *(Affine Conditioning). Let X be any affine source on $\{0,1\}^n$. Let $L : \{0,1\}^n \to \{0,1\}^m$ be any linear function. Then there exist independent affine sources A, B such that:*

- $X = A + B$.
- *For every $b \in \mathsf{Supp}(B)$, $L(b) = 0$.*
- $H(A) = H(L(A))$ *and there exists an affine function $L^{-1} : \{0,1\}^m \to \{0,1\}^n$ such that $A = L^{-1}(L(A))$.*

Proof. Without loss of generality, assume the support of X is a linear subspace (if not, we can do the analysis for the corresponding linear subspace, and then add the affine shift). Consider the set $\{x \in \mathsf{Supp}(X) : L(x) = 0\}$. Note that this set is a linear subspace. Let B be the affine source whose support is this subspace and let $b_1, ..., b_t$ be a basis for this subspace. Next we complete the basis to get a basis for the support of X. Let A be the affine source whose support is the span of the basis vectors in the completed basis that are not in B. Thus $X = A + B$.

Note that $H(A) \leq H(L(A))$ since $L(a) \neq 0$ for every $a \in \mathsf{Supp}(A)$. On the other hand, since L is a deterministic function we have $H(L(A)) \leq H(A)$. Thus $H(A) = H(L(A))$. In other words, L is a bijection between $\mathsf{Supp}(A)$ and $\mathsf{Supp}(L(A))$. Let $Y = L(A)$. Since A is an affine source there exists a vector $a \in \{0,1\}^n$ such that $A = a + \bar{A}$ where $\bar{A}$ is the uniform distribution over a linear subspace. Thus

$$Y = L(A) = L(a) + L(\bar{A}).$$

Let $\bar{Y} = Y - L(a) = L(\bar{A})$. Since L is a linear function and $\bar{A}$ is uniform distributed over a linear subspace, $\bar{Y}$ is also uniformly distributed over a linear subspace. Note that $H(\bar{Y}) = H(Y) = H(L(A)) = H(L(\bar{A}))$, thus L is a linear bijection between the linear subspaces of $\mathsf{Supp}(\bar{A})$ and $\mathsf{Supp}(\bar{Y})$. Therefore there exists a linear function L' such that $\bar{A} = L'(\bar{Y})$. Thus

$$A = a + \bar{A} = a + L'(\bar{Y}) = a + L'(Y - L(a)) = L'(Y) + a - L'(L(a)).$$

Take L^{-1} to be the affine function $L' + a - L'(L(a))$. Then $A = L^{-1}(L(A))$. ■

Now we have the following lemma that exhibits a nice structure of affine sources.

Lemma 2.11. *Let X be any affine source on $\{0,1\}^n$. Divide X into t arbitrary blocks $X = X_1 \circ X_2 \circ ... \circ X_t$. Then there exist positive integers $k_1, ..., k_t$ such that,*

- $\forall j, 1 \leq j \leq t$ *and* $\forall (x_1, .., x_{j-1}) \in \mathsf{Supp}(X_1, .., X_{j-1})$, $H(X_j|_{X_1=x_1,...,X_{j-1}=x_{j-1}}) = k_j$.
- $\sum_{i=1}^{t} k_i = H(X)$.

Proof. For any $i, 1 \leq i \leq t$, let $Y_i = X_1 \circ X_2 \circ ... \circ X_i$. Note Y_i is a linear function of X, thus Y_i is also an affine source. Now for any j, note that Y_{j-1} is a linear function L_j of Y_j. Thus by Lemma 2.10, there exist independent affine sources A_j, B_j such that $Y_j = A_j + B_j$, $H(L_j(Y_j)) = H(A_j)$ and for every $b \in \mathsf{Supp}(B)$, $L_j(b) = 0$. This implies that $Y_{j-1} = L_j(Y_j) = L_j(A_j + B_j) = L_j(A_j)$. Now

since $H(L_j(Y_j)) = H(A_j)$, we have that $\forall (x_1, .., x_{j-1}) \in \mathsf{Supp}(X_1, .., X_{j-1})$, there exists a unique $a_j \in \mathsf{Supp}(A_j)$ such that $L_j(a_j) = (x_1, .., x_{j-1})$.

Let $k_j = H(B_j)$. Then,

$$H(X_j|_{X_1=x_1,...,X_{j-1}=x_{j-1}}) = H(X_j|_{A_j=a_j}) = H(Y_j|_{A_j=a_j}) = H(A_j + B_j|_{A_j=a_j}) = H(B_j) = k_j$$

where the last equality holds because A_j, B_j are independent.

Next, observe that $H(Y_j) = H(A_j) + H(B_j) = H(L_j(Y_j)) + H(B_j) = H(Y_{j-1}) + k_j$. A simple induction thus gives that

$$\sum_{i=1}^{t} k_i = H(X).$$

■

This lemma essentially says that if we divide an affine source into several blocks, then a block has the same entropy conditioned on any fixing of the previous blocks. Moreover, the sum of these entropies equals the entropy of the original source. Thus we can view each block as carrying some fixed additional entropy, regardless of what the previous blocks are. We note that this is very different from general weak random sources.

2.5 Previous Work that We Use

Lemma 2.12. *[ILL89][Leftover Hash Lemma] For any $0 < k < n$, let X be an (n, k)-source and R be the uniform distribution on $\{0,1\}^n$ independent of X. Let $l > 0$ and $m = k - 2l$. Treat x and r as elements in the field F_{2^n} and define the function $\mathsf{Hash}(x, r)$ to be the last m bits of $x \cdot r$. Then $(\mathsf{Hash}(X, R), R)$ is 2^{-l}-close to uniform.*

We are going to use a simple linear merger given in [LRVW03].

Construction 2.13. Let n, s be integers. Define the function

$$\mathsf{Merg} : (\{0,1\}^n)^s \times \{0,1\}^d \to \{0,1\}^n$$

in the following way: Let $\mathbb{F}_q$ be a finite field with q elements where q is a power of 2. Map each element in $\{0,1\}^n$ into $\mathbb{F}_q^\ell$ and each element in $\{0,1\}^d$ into $\mathbb{F}_q^s$, using some injective mapping. Let $x = (x_1, \cdots, x_s) \in (\mathbb{F}_q^\ell)^s$ and $z = (z_1, \cdots, z_s) \in \mathbb{F}_q^s$. The value of $\mathsf{Merg}(x, z)$ is computed as

$$\mathsf{Merg}(x, z) = \sum_{i=1}^{s} x_i \cdot z_i$$

where the operations are performed in the vector space $\mathbb{F}_q^\ell$.

The following theorem is proved in [LRVW03], by using a field of size $O(1/\epsilon)$ in the above construction.

Theorem 2.14. *[LRVW03] For every $\epsilon > 0$ and integers n, s, there exists an explicit (m, ϵ)-merger of (n, s)-somewhere-random sources $\mathsf{Merg} : (\{0,1\}^n)^s \times \{0,1\}^d \to \{0,1\}^n$ with $d = O(s \log(1/\epsilon))$ and $m = n/2 - O(d)$. Moreover, for any (n, s)-somewhere-random source X, with probability $1 - O(\epsilon)$ over $z \in \{0,1\}^d$, $\mathsf{Merg}(X, z)$ is ϵ-close to having min-entropy m.*

We are going to use condensers recently constructed based on the sum-product theorem. The following construction is due to Zuckerman [Zuc07].

Construction 2.15. Let $F = \mathbb{F}_q$ be a field where $q = 2^p$ for p prime. Define the point-line incidence graph as the bipartite graph $G = (V, W, E)$ with vertices $V = F^2$ the set of points, and W the set of lines over F, and (p, l) is an edge in G iff p and l are incident. Let the function $h : E \to V \times W$ map an edge to its two endpoints. Equivalently, h is the map from F^3 to $(F^2)^2$ such that $h(a, b, c) = ((b, ab + c), (a, c))$.

The condenser $C : F^3 \times \{0,1\} \to F^2$ is $C(e, i) = h(e)_i$.

The following theorem is proved in [Zuc07].

Theorem 2.16. *[Zuc07] Suppose $\delta < 0.9$ and $q^\delta = \omega(1)$. The function C above is a rate-$(\delta \to (1 + \alpha/2)\delta, \epsilon)$-somewhere-condenser, where $\epsilon = q^{-\alpha\delta/20}$ for some constant $\alpha > 0$.*

Note that each bit of the output of the condenser is a degree 2 polynomial of the bits of the input. Repeating the condenser for a constant number of times, we get the following theorem:

Theorem 2.17. *[Zuc07] For any constant $\beta, \delta > 0$, there is an efficient family of rate-$(\delta \to 1 - \beta, \epsilon = 2^{-\Omega(n)})$-somewhere condensers $\mathrm{Zuc} : \{0,1\}^n \to (\{0,1\}^m)^D$ where $D = O(1)$ and $m = \Omega(n)$. Moreover, each bit of the output is a constant degree polynomial of the bits of the input.*

We now show that this condenser actually works even when the min-entropy of the source is very high. First we need the following improved theorem about line point incidences in finite fields.

Theorem 2.18 (Incidence Theorem)**.** *[Vin07] Let $F = \mathbb{F}_q$, where q is either prime or 2^p for p prime. Let P, L be sets of points and lines in F^2 and $|P|, |L| \le N = q^\alpha$ with $1 + \gamma \le \alpha \le 2 - \gamma$ for some $\gamma > 0$. Then the number of incidences*

$$I(P, L) \le 2N^{\frac{3}{2} - \frac{\gamma}{4}}.$$

The following lemma is from [Zuc07].

Lemma 2.19. *[Zuc07] If X, Y is not ϵ-close to a somewhere-k-source, then there exists sets $S \subseteq \mathsf{supp}(X), T \subseteq \mathsf{supp}(Y), |S|, |T| < 2^{k+1}/\epsilon$, such that*

$$\Pr[X \in S, Y \in T] > \epsilon/2.$$

Theorem 2.20. *Suppose $\frac{1}{2} < \delta \le 1 - \gamma$ for some $\gamma > 0$. The function C above is a rate-$(\delta \to (1 + \gamma/12)\delta, \epsilon)$ somewhere-condenser, where $\epsilon = q^{-\gamma\delta/20}$.*

Proof. We essentially follow the proof in [Zuc07]. As in that proof, we analyze the equivalent function h. We may assume that the input to h is uniform on a set of edges of size $K = 2^k = q^{3\delta}$, and set $k' = (1 + \gamma/12)(2k/3)$. Suppose the output (X, Y) of h is not ϵ-close to a somewhere-k'-source. Let $P = S$ and $L = T$ be the sets of size less than $K_0 = 2^{k'+1}/\epsilon$ given by Lemma 2.19. Note that $K_0 \le 2q^{2\delta(1+\gamma/12)+\delta(\gamma/20)} < q^{2-\gamma}$.

Now we calculate the number of incidences $I(P,L)$ in two ways. On the one hand, since each edge is an incident point-line pair, and at least $\epsilon/2$ fraction of these pairs lie in $P \times L$, the number of incidences $I(P,L) \geq \epsilon K/2$. On the other hand, by Theorem 2.18,

$$I(P,L) \leq 2K_0^{3/2-\gamma/4} = O(K^{(1+\gamma/12)(3/2-\gamma/4)2/3}/\epsilon^2) = O(K^{1-\gamma/12}/\epsilon^2).$$

This gives a contradiction for $\epsilon = K^{-\gamma/60}$, and the theorem is proved. ■

Our affine extractor can use any affine extractor for entropy rate $1/2$ as a black box, as long as the extractor is a low degree polynomial. Currently, there are two such constructions, one is a degree 3 polynomial and the other is a degree 6 polynomial. For example, we can use Bourgain's simple affine extractor for entropy rate slightly below $1/2$.

Theorem 2.21. *[Bou07] There is a polynomial time computable function* $\mathsf{BAExt} : \{0,1\}^n \to \{0,1\}^m$ *such that* $m = \Omega(n)$ *and for every affine source* X *of entropy* $n/2$, $\mathsf{BAExt}(X)$ *is* $2^{-\Omega(n)}$*-close to uniform. Moreover, each bit of the output is a degree 3 polynomial of the bits of the input.*

We need one last ingredient, the simple inner product function as a two source extractor when the sum of the entropy rates of the two independent sources is greater than 1. For a finite field $\mathbb{F}$, let $\mathsf{Had} : \mathbb{F}^l \times \mathbb{F}^l \to \mathbb{F}$ be the inner product function, i.e., $\mathsf{Had}(x,y) = x \cdot y$.

Theorem 2.22. *[CG88, Vaz85] For every constant* $\delta > 0$*, there exists a polynomial time algorithm* $\mathsf{Had} : (\{0,1\}^n)^2 \to \{0,1\}^m$ *such that if* X *is an* (n, k_1) *source,* Y *is an independent* (n, k_2) *source and* $k_1 + k_2 \geq (1+\delta)n$*, then*

$$|(Y, \mathsf{Had}(X,Y)) - (Y, U_m)| < \epsilon$$

with $m = \Omega(n)$ *and* $\epsilon = 2^{-\Omega(n)}$.

To prove our construction is an extractor, we need the following definition and lemma.

Definition 2.23. (ϵ-biased space) A random variable Z over $\{0,1\}$ is ϵ-biased if $|\Pr[Z=0] - \Pr[Z=1]| \leq \epsilon$. A sequence of 0-1 random variables $Z_1, \cdots, Z_m$ is *ϵ-biased for linear tests* if for any nonempty set $S \subset \{1, \cdots, m\}$, the random variable $Z_S = \bigoplus_{i \in S} Z_i$ is ϵ-biased.

The following lemma is due to Vazirani. For a proof see for example [Gol95]

Lemma 2.24. *Let* $Z_1, \cdots, Z_m$ *be 0-1 random variables that are* ϵ*-biased for linear tests. Then, the distribution of* $(Z_1, \cdots, Z_m)$ *is* $\epsilon \cdot 2^{m/2}$*-close to uniform.*

3 Low Degree Strong Linear Seeded Extractors

In this section we describe our construction of a strong linear seeded extractor. The extractor has the property that each bit of the output is a degree 4 polynomial of the bits of the input.

Theorem 3.1. *There exists a constant* $0 < \beta < 1$ *such that for every* $0 < \delta < 1$ *and any* $1/\sqrt{n} < \alpha < 1$ *there exists a polynomial time computable function* $\mathsf{LSExt} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *s.t.*

- $d \leq \alpha n, m \geq \beta\delta\alpha n$.

- *For any $(n, \delta n)$-affine source [1] X, let R be the uniform distribution on $\{0,1\}^d$ independent of X. Then $(\mathsf{LSExt}(X,R),R)$ is $2^{-\Omega(\delta\alpha^2 n)}$-close to uniform.*
- *Each bit of the output is a degree 4 polynomial of the bits of the two inputs, and for any fixing of r the output is a linear function of x.*

Proof. The construction of the extractor consists of 3 steps:

Step 1: We take $d_1 = \alpha n/3$ random bits R_1 and divide X into $3/\alpha$ blocks of equal length[2] $X = X_1 \circ \cdots \circ X_t$, where each X_i has $\alpha n/3$ bits. We now apply the function Hash as in Lemma 2.12 to every X_i and R_1 and concatenate the output to be $Y = Y_1 \circ \cdots \circ Y_t$, where $Y_i = \mathsf{Hash}(X_i, R_1)$. We let each Y_i output $l_1 = 0.9\delta\alpha n/3 = 0.3\delta\alpha n$ bits.

Claim 3.2. *With probability $1 - 2^{-\Omega(\delta\alpha n)}$ over R_1, Y is $2^{-\Omega(\delta\alpha n)}$-close to being a $(l_1, 3/\alpha)$-somewhere random source.*

Proof of the claim. By Lemma 2.11 there exist integers $k_1, \cdots, k_t$ such that for any fixing of the previous blocks, X_i has entropy k_i. Note $\sum_{i=1}^{t} k_i = \delta n$. Thus there exists $1 \le i \le t$ such that $k_i \ge \delta\alpha n/3$. Now by Lemma 2.12 we know that (Y_i, R_1) is $2^{-\Omega(\delta\alpha n)}$-close to (U, R_1). Therefore with probability $1 - 2^{-\Omega(\delta\alpha n)}$ over R_1, Y_i is $2^{-\Omega(\delta\alpha n)}$-close to uniform. Thus Y is $2^{-\Omega(\delta\alpha n)}$-close to being a $(l_1, 3/\alpha)$-somewhere random source. ∎

Step 2: Let $\epsilon = 2^{-c\delta\alpha^2 n}$ for a constant $0 < c < 1$ to be chosen later. We take d_2 random bits R_2 and apply the merger as in Theorem 2.14 with parameter ϵ. Let $W = \mathsf{Merg}(Y, R_2)$.

Claim 3.3. *$d_2 \le \alpha n/3$ and with probability $1 - 2^{-\Omega(\delta\alpha^2 n)}$ over R_2, W is $2^{-\Omega(\delta\alpha^2 n)}$-close to having min-entropy $0.4l_1$.*

Proof of the claim. By Theorem 2.14 the seed length $d_2 = O(c\delta\alpha^2 n/\alpha) = O(c\delta\alpha n)$. Note that $l_1 = 0.3\delta\alpha n$. Thus we can choose c s.t. $d_2 \le \alpha n/3$ and the output of the merger has length $m = l_1/2 - O(d_2) \ge 0.4l_1$. Now by Theorem 2.14 we know that with probability $1 - 2^{-\Omega(\delta\alpha^2 n)}$ over R_2, W is $2^{-\Omega(\delta\alpha^2 n)}$-close to having min-entropy $0.4l_1$. ∎

Step 3: Now W is a random variable over $l_1 < \alpha n/3$ bits. Take $d_3 = l_1$ random bits R_3 and apply the function Hash as in Lemma 2.12 to W and R_3 and output $m = 0.3l_1$ bits. The final output is $Z = \mathsf{Hash}(W, R_3)$.

The number of random bits used is $d = d_1 + d_2 + d_3 \le \alpha n$. The number of bits of the output is $m = 0.3l_1 \ge \beta\delta\alpha n$ for some constant $0 < \beta < 1$. By Lemma 2.12 with probability $1 - 2^{-\Omega(\delta\alpha n)}$ over R_3, Z is $2^{-\Omega(\delta\alpha n)}$-close to uniform. Thus with probability $1 - 2^{-\Omega(\delta\alpha^2 n)}$ over R, Z is $2^{-\Omega(\delta\alpha^2 n)}$-close to uniform, which implies that $(\mathsf{LSExt}(X,R),R)$ is $2^{-\Omega(\delta\alpha^2 n)}$-close to uniform.

In each of the 3 steps the degree of the polynomial goes up by 1. Thus each bit of the output is a degree 4 polynomial of the bits of the two inputs. Now observe in each step for any fixing of

[1] Generally we don't need the source to be affine. However the analysis is simpler if the source is an affine source (mainly because in this case we don't need to go into convex combinations as in the case where the source is a general weak source).

[2] For simplicity, we assume that $3/\alpha$ is an integer, this does't affect the analysis.

$R_i = r$ the output is a linear function, thus for any fixing of $R = r$ the output is a linear function of x. ■

In the special case where α and δ are constants, we get the following corollary:

Corollary 3.4. *For all constants $0 < \delta, \alpha < 1$ there exists a polynomial time computable function $\mathsf{LSExt} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ and a constant $0 < \beta < 1$ such that*

- *$d \leq \alpha n, m \geq \beta n$.*
- *For any $(n, \delta n)$-affine source X, let R be the uniform distribution on $\{0,1\}^d$ independent of X. Then $(\mathsf{LSExt}(X,R), R)$ is $2^{-\Omega(n)}$-close to uniform.*
- *Each bit of the output is a degree 4 polynomial of the bits of the two inputs, and for any fixing of r the output is a linear function of x.*

4 Extractors for Affine Somewhere Random Sources with Few Rows

In this section we describe our extractor for an affine somewhere random source with few rows. The construction is essentially the same as that in [Rao09], except that we use our low degree strong linear seeded extractor in the construction.

We need the following definition about the slice of a concatenation of strings.

Definition 4.1. [Rao09] Given ℓ strings of length n, $x = x_1, \cdots, x_\ell$, define $\mathsf{Slice}(x, w)$ to be the string $x' = x'_1, \cdots, x'_\ell$ such that for each i x'_i is the prefix of x_i of length w.

Algorithm 4.2 ($\mathsf{AffineCondense}(x)$)**.**

Input: x — a $t \times r$ matrix with $t < \sqrt[4]{r}$.
Output: y — a $\lceil t/2 \rceil \times m$ matrix with $m = \Omega(r/t^2)$.

Sub-Routines and Parameters:
Let $w = r/(10t)$.
Let $\mathsf{BAExt} : \{0,1\}^n \to \{0,1\}^d$ be the affine extractor from Theorem 2.21.
Let $\mathsf{LSExt} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ be the strong linear seeded extractor from Theorem 3.1.

1. Let z be the $\lceil t/2 \rceil \times 2w$ matrix obtained by concatenating pairs of rows in $\mathsf{Slice}(x, w)$, i.e., the i'th row z_i is $\mathsf{Slice}(x,w)_{2i-1} \circ \mathsf{Slice}(x,w)_{min\{2i,t\}}$.
2. Let s be the $\lceil t/2 \rceil \times d$ matrix whose i'th row is $\mathsf{BAExt}(z_i)$.
3. Let y be the $\lceil t/2 \rceil \times m$ matrix whose i'th row is $\mathsf{LSExt}(x, s_i)$.

Algorithm 4.3 ($\mathsf{AffineSRExt}(x)$).

Input: x — a $t \times r$ matrix.
Output: z — an m bit string with $m = r/t^{O(\log t)}$.

Sub-Routines and Parameters:

1. If x has one row, output x.
2. Else, set y to be the output of $\mathsf{AffineCondense}(x)$.
3. Set $x = y$ and go to the first step.

Lemma 4.4. *For any $t \times r$ affine somewhere random source X with $t < \sqrt[4]{r}$, $\mathsf{AffineCondense}(X)$ is $2^{-\Omega(r/t^4)}$-close to a convex combination of $\lceil t/2 \rceil \times m$ affine somewhere random sources, where $m = \Omega(r/t^2)$. Moreover, each bit of the output is a constant degree polynomial of the input bits.*

Proof. We essentially follow the proof in [Rao09], except that we use the specific strong linear seeded extractor LSExt.

Let $Z = \mathsf{Slice}(X, w)$ as in the algorithm. Note that $\mathsf{Slice}(X, w)$ is a linear function of X. Thus by Lemma 2.10, there exist independent affine sources A and B s.t. $X = A + B$, $H(A) = H(\mathsf{Slice}(A, w))$ and for every $b \in \mathsf{Supp}(B)$, $\mathsf{Slice}(b, w) = 0$. This implies that $Z = \mathsf{Slice}(X, w) = \mathsf{Slice}(A, w) + \mathsf{Slice}(B, w) = \mathsf{Slice}(A, w)$ is independent of B and $H(B) = H(X) - H(A) = H(X) - H(\mathsf{Slice}(X, w)) \geq r - wt$.

Note that Z is a linear function of X, thus conditioned on any fixing $Z = z$, $X|Z = z$ is an affine source. Moreover, conditioned on any fixing $Z = z$, Y is a linear function of $X|Z = z$ (because LSExt is a linear seeded extractor). Thus conditioned on any fixing $Z = z$, $Y|Z = z$ is affine. We next show that with high probability over $z \leftarrow_R Z$, $Y|Z = z$ is somewhere random.

Since X is somewhere random, there exists an index h s.t. Z_h is an affine source with entropy rate $1/2$. Therefore by Theorem 2.21, s_h has $\Omega(w)$ bits and

$$|S_h - U_d| \leq 2^{-\Omega(w)}. \tag{1}$$

Note that S_h is a deterministic function of Z and is thus independent of B. Also s_h has $d = \Omega(w) = \Omega(r/t) = \Omega(1/t^2 \cdot tr)$ bits. Note that B has tr bits and $H(B) \geq r - wt \geq 0.9r$. Let U_d be the uniform distribution over d bits independent of B. Then by Theorem 3.1 we have that $\mathsf{LSExt}(B, S_h)$ has $m = \Omega(1/t^3 \cdot tr) = \Omega(r/t^2)$ bits and

$$\Pr_{u \leftarrow_R U_d}[|\mathsf{LSExt}(B, u) - U_m| > \epsilon] < \epsilon$$

where $\epsilon = 2^{-\Omega(r/t^4)}$.

By Proposition 2.9 and equation 1 we thus have

$$\Pr_{s \leftarrow_R S_h}[|\mathsf{LSExt}(B, s) - U_m| > 0] < \epsilon + 2^{-\Omega(w)} = 2^{-\Omega(r/t^4)}.$$

For any $s \in \{0,1\}^d$, we have $\mathsf{LSExt}(X,s) = \mathsf{LSExt}(A+B,s) = \mathsf{LSExt}(A,s) + \mathsf{LSExt}(B,s)$. Note that S_h is a deterministic function of Z, Z is a deterministic function of A and $H(A) = H(Z)$. Thus A is also completely determined by Z. Therefore whenever $\mathsf{LSExt}(B,s)|Z = z$ is uniform, $\mathsf{LSExt}(X,s)|Z = z$ is also uniform.

Thus we get

$$\Pr_{z \leftarrow_R Z}[|\mathsf{LSExt}(X|Z = z, s_h) - U_m| > 0] \leq 2^{-\Omega(r/t^4)}.$$

This shows that Y is $2^{-\Omega(r/t^4)}$-close to being a convex combination of affine somewhere random sources. Now note that both BAExt and LSExt are constant degree polynomials, thus each bit of the output is a constant degree polynomial of the input bits ■

Repeating the condenser for $\log t$ times, we get the following theorem:

Theorem 4.5. *For every affine $t \times r$ somewhere random source X, $\mathsf{AffineSRExt}(X)$ outputs $m = r/t^{O(\log t)}$ bits that are $2^{-\Omega(r/t^{O(\log t)})}$-close to uniform. Moreover, each bit of the output is a degree $t^{O(1)}$ polynomial of the bits of the input.*

In the special case where t is a constant, we get the following corollary.

Corollary 4.6. *For every affine $t \times r$ somewhere random source X with $t = O(1)$, $\mathsf{AffineSRExt}(X)$ outputs $m = \Omega(r)$ bits that are $2^{-\Omega(r)}$-close to uniform. Moreover, each bit of the output is a constant degree polynomial of the bits of the input.*

5 The Main Construction

We now describe our main constructions. For simplicity we first describe our constructions for affine sources with linear entropy, we then briefly show how we can generalize the constructions to handle slightly sub-linear entropy.

First we describe our construction of an affine disperser.

5.1 Affine Dispersers for Linear Entropy Sources

Given an n-bit affine source X with entropy δn for some constant $\delta > 0$, we first divide X into $10/\delta$ blocks of equal size $X = X_1 \circ \cdots \circ X_t$, where $t = 10/\delta$ and each block has $\delta n/10$ bits. The algorithm is described as follows.

Algorithm 5.1 ($\mathsf{ADisp}(x)$)**.**

Input: x — an n bit string.
Output: z — an m bit string with $m = \Omega(n)$.

Sub-Routines and Parameters:
Let $\mathrm{Zuc} : \{0,1\}^n \to (\{0,1\}^{\Omega(n)})^{O(1)}$ be a rate-$(\delta/4 \to 1-\delta/4, 2^{-\Omega(n)})$-somewhere-condenser form Theorem 2.17.
Let $\mathsf{Had} : (\{0,1\}^n)^2 \to \{0,1\}^{\Omega(n)}$ be the two-source extractor from Theorem 2.22, set up to extract from two independent sources whose entropy rates sum up to more than $1+\delta/4$.
Let $\mathsf{LSExt} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{m'}$ be the strong linear seeded extractor from Theorem 3.1.
Let $\mathsf{AffineSRExt}$ be the extractor for affine somewhere random sources from Algorithm 4.3.

Divide x into $10/\delta$ blocks $x = x_1 \circ \cdots \circ x_t$ where $t = 10/\delta$ and each block has $\delta n/10$ bits.
For every $i, 1 \le i \le t$ do the following.

1. Let $y_{i1} \circ \cdots \circ y_{i\ell_1} = \mathrm{Zuc}(x_i)$, where y_{ij} is the j'th row of the matrix obtained by applying Zuc to x_i. Note $\ell_1 = O(1)$ and each y_{ij} has $\Omega(n)$ bits.
2. Divide x into ℓ_2 blocks of equal size $x = x'_1 \circ \cdots \circ x'_{\ell_2}$, with each block having the same number of bits as y_{ij}. Note $\ell_2 = O(1)$. Apply Had to every pair of x'_{j_2} and y_{ij_1}, and output $\delta^3 n/(3000\ell_1\ell_2)$ bits. Let sr_i be the matrix obtained by concatenating all the outputs $\mathsf{Had}(x'_{j_2}, y_{ij_1})$, i.e., each row of sr_i is $\mathsf{Had}(x'_{j_2}, y_{ij_1})$ for a pair (x'_{j_2}, y_{ij_1}).
3. Let $r_i = \mathsf{AffineSRExt}(sr_i)$.
4. Let $u_i = \mathsf{LSExt}(x, r_i)$, set up to output at most $\delta^3 n/(3000\ell_1\ell_2)$ bits.
5. Divide the bits of u_i into $s_i = \Omega(n)$ blocks of equal size, with each block having c_i number of bits, for some constant c_i to be chosen later. For every $j = 1, \cdots, s_i$, compute one bit v_{ij} by taking the product of all the bits in the j'th block, i.e., $v_{ij} = \Pi_{\ell=(j-1)c_i+1}^{jc_i} u_{i\ell}$.

Finally, output $\Omega(n)$ bits $\{z_j = \bigoplus_{i=1}^t v_{ij}\}$.

Theorem 5.2. *For every $\delta > 0$ there exists an efficient family of functions $\mathsf{ADisp} : \{0,1\}^n \to \{0,1\}^m$ such that $m = \Omega(n)$ and for every affine source X with entropy δn, $|\mathsf{Supp}(\mathsf{ADisp}(X))| = 2^m$.*

Proof. We show that Algorithm 5.1 is an efficient family of such functions. First, by Lemma 2.11, when we divide X into $t = 10/\delta$ blocks of equal size, there exist positive integers $k_1, \cdots, k_t$ s.t. for any fixing of the previous blocks, $H(X_i) = k_i$ and $\sum_{i=1}^t k_i = \delta n$. Thus there must exist an i s.t. $k_i \ge \delta n/3t$. Let X_g be the first such block, i.e., g is the smallest i s.t. $k_i \ge \delta n/3t$.

Lemma 5.3. *Conditioned on any fixing of $(X_i = x_i, SR_i = sr_i, R_i = r_i, U_i = u_i)_{i\in\{1,\cdots,g-1\}}$, X is an affine source with $H(X_g) \ge \delta n/4t$ and $H(X) \ge 3\delta n/5$.*

Proof of the lemma. We first fix $(X_i = x_i)_{i\in\{1,\cdots,g-1\}}$. Note since X_i is a linear function of X, after this fixing X is still an affine source. Now by Lemma 2.11, after this fixing $H(X_g) \ge k_g \ge \delta n/3t$ and $H(X) \ge \sum_{i=g}^t k_i \ge \delta n - t \cdot \delta n/3t \ge 2\delta n/3$.

Note that conditioned on the fixing of $(X_i = x_i)_{i\in\{1,\cdots,g-1\}}$, SR_i is a linear function of X. Thus we can further fix $(SR_i = sr_i)_{i\in\{1,\cdots,g-1\}}$ and X is still an affine source. Note that SR_i has $\ell_1\ell_2$ rows and each row has $\delta^3 n/(3000\ell_1\ell_2)$ bits, thus SR_i has a total number of $\delta^3 n/3000$ bits. Let $\overline{SR} = SR_1 \circ \cdots \circ SR_{g-1}$ and abuse notation to let $\overline{SR}(X)$ stand for the linear function of X that computes $\overline{SR}$. Note $\overline{SR}$ has at most $\delta^3 n/3000 \cdot t = \delta^2 n/300$ bits. By Lemma 2.10, there exist independent affine sources A and B s.t. $X = A + B$, $\overline{SR}(X) = \overline{SR}(A)$ and $H(\overline{SR}(A)) = H(A)$. Thus conditioned on any fixing of $\overline{SR}$,

$$H(X) \geq H(B) \geq 2\delta n/3 - H(A) = 2\delta n/3 - H(\overline{SR}(X)) \geq 2\delta n/3 - \delta^2 n/300.$$

Next note that X_g is a linear function of X. Thus $X_g(X) = X_g(A) + X_g(B)$. Therefore $H(X_g(B)) = H(X_g) - H(X_g(A)) \geq H(X_g) - H(A) \geq \delta n/3t - \delta^2 n/300$. Since $\overline{SR}(X) = \overline{SR}(A)$ we have that conditioned on any fixing of $\overline{SR}(X)$, $H(X_g) \geq H(X_g(B)) \geq \delta n/3t - \delta^2 n/300$.

Note that after the fixing of $(SR_i = sr_i)_{i\in\{1,\cdots,g-1\}}$, $(R_i)_{i\in\{1,\cdots,g-1\}}$ is also fixed, and now $(U_i)_{i\in\{1,\cdots,g-1\}}$ is a linear function of X. Thus by the same analysis we have that conditioned on any fixing of $(U_i)_{i\in\{1,\cdots,g-1\}}$, X is still an affine source. Moreover, $H(X) \geq 2\delta n/3 - \delta^2 n/300 - \delta^2 n/300 > 3\delta n/5$ and $H(X_g) \geq \delta n/3t - \delta^2 n/300 - \delta^2 n/300 > \delta n/4t$. ∎

Now consider X and X_g conditioned on any fixing of $(X_i = x_i, SR_i = sr_i, R_i = r_i, U_i = u_i)_{i\in\{1,\cdots,g-1\}}$, we have the following lemma.

Lemma 5.4. *With probability $1 - 2^{-\Omega(n)}$ over the further fixings of $X_g = x_g$, R_g is $2^{-\Omega(n)}$-close to uniform.*

Proof of the lemma. First, note that $H(X_g) \geq \delta n/4t$, thus X_g has entropy rate at least $\delta/4$. Therefore by Theorem 2.17 $\mathrm{Zuc}(X_g)$ is $2^{-\Omega(n)}$-close to a somewhere-rate-$(1-\delta/4)$ source. Without loss of generality assume that Y_{g1} has rate $1 - \delta/4$. Since X_g is a linear function of X, by Lemma 2.10 there exist independent affine sources A_g and B_g such that $X = A_g + B_g$, $X_g(X) = X_g(A_g)$ and $H(A_g) = H(X_g)$. Thus $H(B_g) = H(X) - H(A_g) = H(X) - H(X_g) \geq 3\delta n/5 - \delta n/10 = \delta n/2$. Note that when we divide X into ℓ_2 blocks $X = X'_1 \circ \cdots \circ X'_{\ell_2}$, each X'_j is a linear function of X. Thus $X'_j(X) = X'_j(A_g) + X'_j(B_g)$. Let $A_{gj} = X'_j(A_g)$ and $B_{gj} = X'_j(B_g)$. Since B_g is an affine source, by Lemma 2.11 the sum of all $H(B_{gj})$ is at least $H(B_g) \geq \delta n/2$. Thus at least one block must have entropy rate at least $\delta/2$. Let B_{gj} be such a block.

Note that Y_{g1} is a deterministic function of X_g, thus it is also a deterministic function of A_g and is independent of B_{gj}. Note Y_{g1} has rate $1 - \delta/4$ and B_{gj} has rate $\delta/2$. Thus by Theorem 2.22 we have that with probability $1 - 2^{-\Omega(n)}$ over the fixings of Y_{g1} (and thus A_g and X_g), $\mathsf{Had}(B_{gj}, Y_{g1})$ is $2^{-\Omega(n)}$-close to uniform. Now note that for a fixed $Y_{g1} = y_{g1}$, the function Had is a linear function. Therefore

$$\mathsf{Had}(X'_j, y_{g1}) = \mathsf{Had}(A_{gj}, y_{g1}) + \mathsf{Had}(B_{gj}, y_{g1}).$$

Note that once A_g (equivalently, X_g) is fixed, $\mathsf{Had}(A_{gj}, y_{g1})$ is a fixed constant. Thus whenever a fixed A_g makes $\mathsf{Had}(B_{gj}, y_{g1})$ uniform, it also makes $\mathsf{Had}(X'_j, y_{g1})$ uniform. Therefore we have that with probability $1 - 2^{-\Omega(n)}$ over the fixings of A_g (and thus X_g), $\mathsf{Had}(X'_j, Y_{g1})$ is $2^{-\Omega(n)}$-close to uniform. When this happens, we have that SR_g is $2^{-\Omega(n)}$-close to an affine somewhere random source (it is affine since for a fixed $X_g = x_g$, SR_g is a linear function of X). Thus by Theorem 4.5 R_g is $2^{-\Omega(n)}$-close to uniform. ∎

Now consider X conditioned on any fixing of $(X_i = x_i, SR_i = sr_i, R_i = r_i, U_i = u_i)_{i\in\{1,\cdots,g-1\}}$, and the event that the property in Lemma 5.4 is satisfied. We have the following lemma.

Lemma 5.5. *With probability $1 - 2^{-\Omega(n)}$ over the further fixings of SR_g (and thus R_g), U_g is uniform.*

Proof of the lemma. First note that X_g is a linear function of X. Thus conditioned on any fixing of $X_g = x_g$, X is still an affine source. Now after we fix $X_g = x_g$, SR_g is a linear function of X. Thus again by Lemma 2.10, there exist independent affine sources A'_g and B'_g s.t. $X = A'_g + B'_g$, $SR_g(X) = SR_g(A'_g)$ and $H(SR_g) = H(A'_g)$. Thus $H(B'_g) = H(X) - H(A'_g) = H(X) - H(SR_g) \geq 3\delta n/5 - \delta n/10 - \delta^3 n/3000 > \delta n/3$.

Next, note R_g is a deterministic function of SR_g and is $2^{-\Omega(n)}$-close to uniform by Lemma 5.4. Thus R_g is independent of B'_g. Note LSExt is a strong linear seeded extractor with error $2^{-\Omega(n)}$ by Corollary 3.4. Thus by Proposition 2.9 with probability $1 - 2^{-\Omega(n)}$ over the fixings of R_g (and thus SR_g), $\mathsf{LSExt}(B'_g, R_g)$ is uniform.

Finally note that for any fixing of SR_g (and thus R_g), $\mathsf{LSExt}(X, R_g)$ is a linear function of X. Thus by the same analysis as in Lemma 5.4 we have that with probability $1 - 2^{-\Omega(n)}$ over the fixings of SR_g (and thus R_g), U_g is uniform. ∎

Now consider X conditioned on any fixing of $(X_i = x_i, SR_i = sr_i, R_i = r_i, U_i = u_i)_{i\in\{1,\cdots,g-1\}}$, and the event that both properties in Lemma 5.4 and Lemma 5.5 hold. Note that even further conditioned on the fixings of $X_g = x_g$ and $SR_g = sr_g$, X is still an affine source. Now after this fixing U_g is a linear function of X. Thus again by Lemma 2.10 there exist independent affine sources A''_g and B''_g s.t. $X = A''_g + B''_g$, $U_g(X) = U_g(A''_g)$ and $H(U_g) = H(A''_g)$. To prove the function ADisp is a disperser, it suffices to prove that for some $B''_g = b$, it is a disperser. We actually can prove that for any $B''_g = b$, ADisp is a disperser.

Lemma 5.6. *For any integer $m > 0$, let $Z' = (Z'_1, \cdots, Z'_m)$ where $Z'_j = \bigoplus_{i=g}^{t} V_{ij}$. Then conditioned on any fixing of $B''_g = b$, $|\mathsf{Supp}(Z')| = 2^m$.*

Proof of the lemma. We first show that Z'_1 can take both values in $\{0, 1\}$. To see this, notice that $Z'_1 = \bigoplus_{i=g}^{t} V_{i1}$ while $V_{g1} = \Pi_{\ell=1}^{c_g} U_{g\ell}$ is a polynomial of degree c_g over the bits of the uniform string U_g. Next, since now (conditioned on all the fixings) U_g is a linear function of X, by Lemma 2.10 there exists an affine function L_g s.t. $A''_g = L_g(U_g)$. Thus $X = A''_g + B''_g = L_g(U_g) + B''_g$. Note $U_g = U_g(X) = U_g(A''_g)$ is independent of B''_g. Now conditioned on any fixing of $B''_g = b$, $X = L_g(U_g) + b$ is an affine function (degree 1 polynomial) of U_g.

Given this observation, the following computations and the outputs $V_{(g+1)1}, \cdots, V_{t1}$ are all functions of U_g. If we can show that $\bigoplus_{i=g+1}^{t} V_{i1}$ is a polynomial of degree less than c_g over the bits of U_g, then we know that $Z'_1 = V_{g1} \oplus \bigoplus_{i=g+1}^{t} V_{i1}$ cannot be a constant and thus must take both values of $\{0, 1\}$. In fact, that is exactly how we choose the constants c_i's.

Let us see what conditions c_i's must satisfy. First, it's easy to see that we need to choose c_i's s.t. $c_i > c_{i+1}$ for every i. Next, we compute the degrees of each V_{i1} for $i > g$. First X is an affine function of U_g. Then by Theorem 2.17, each bit of $\mathrm{Zuc}(X_i)$ is a constant degree polynomial of the input bits. It's easy to see the function Had is a degree 2 polynomial. Thus each bit of SR_i is a degree 2 polynomial of the input bits. By Corollary 4.6 each bit of R_i is a constant degree polynomial of the input bits. By Theorem 3.1 each bit of U_i is a constant degree polynomial of the input bits. Thus we conclude that for every $i \geq g + 1$, each bit of U_i is a constant $c(\delta)$-degree

polynomial of the bits of U_g. Note each V_{i1} is a degree c_i polynomial of the bits of U_i. Therefore in order to make $\bigoplus_{i=g+1}^{t} V_{i1}$ a polynomial of degree less than c_g, it suffices to take

$$c_i > c(\delta)c_{i+1}$$

for every i. This ensures that Z'_1 can take both values in $\{0,1\}$.

Now we consider outputting $m > 1$ bits. By induction it suffices to prove that for any fixing of $(Z'_1 = z_1, \cdots, Z'_i = z_i)$, Z'_{i+1} can take both values in $\{0,1\}$. For the sake of contradiction, suppose for some $(Z'_1 = z_1, \cdots, Z'_i = z_i)$, Z'_{i+1} can only take the value z_{i+1}. Then we have

$$P_g = (Z'_1 + z_1 + 1)(Z'_2 + z_2 + 1)\cdots(Z'_i + z_i + 1)(Z'_{i+1} + z_{i+1}) \equiv 0.$$

However, note that $Z'_j = V_{gj} \oplus \bigoplus_{i=g+1}^{t} V_{ij}$ and V_{gj} is a monomial of degree c_g of the bits of U_g, while all the other V_{ij}'s are monomials of degree less than c_g of the bits of U_g. Also note that V_{gj}'s are monomials of different bits for different j's. Thus P_g is a polynomial of the bits of U_g and has one monomial of degree $(i+1)c_g$ (the highest degree monomial) with coefficient 1. Therefore P_g cannot always be 0. ■

Note that once $(X_i = x_i, SR_i = sr_i, R_i = r_i, U_i = u_i)_{i\in\{1,\cdots,g-1\}}$ are fixed, $(V_{ij})_{i\in\{1,\cdots,g-1\}}$ are also fixed. The theorem now follows immediately from Lemma 5.3, Lemma 5.4 and Lemma 5.5. Note that since $t = O(1)$ we can satisfy the conditions $c_i > c(\delta)c_{i+1}$ while keeping all c_i's to be constants. Thus the disperser can output $\Omega(n)$ bits. ■

5.2 Affine Extractors for Linear Entropy Sources

With the affine disperser in hand, the affine extractor is just one step away. We first show how to extract 1 bit that is $2^{-\Omega(n)}$-close to uniform. For this we need the following definition and theorem.

Definition 5.7. For two functions $f, p : \{0,1\}^n \to \{0,1\}$, their correlation over the uniform distribution is defined as

$$\mathsf{Cor}(f,p) = \left|\Pr_x[f(x) = p(x)] - \Pr_x[f(x) \neq p(x)]\right|,$$

where the probability is over the uniform distribution. For a class C of functions, we denote by $\mathsf{Cor}(f,C)$ the maximum of $\mathsf{Cor}(f,p)$ over all functions $p \in C$ whose domain is $D := dom(f)$.

Theorem 5.8. *(XOR lemma for polynomials over* GF(2)*) [VW08, BKS+09] Let P_d stand for the class of all polynomials of degree at most d over* GF(2)*. Let $f : \{0,1\}^n \to \{0,1\}$ be a function such that $\mathsf{Cor}(f, P_d) \leq 1 - 1/2^d$ and $f^{\times m}$ be the XOR of the value of f on m independent inputs. Then*

$$\mathsf{Cor}(f^{\times m}, P_d) \leq \exp(-\Omega(m/(4^d \cdot d))).$$

Theorem 5.9. *Let $Z = (Z_1, \cdots, Z_m)$ be the output of the affine disperser in Algorithm 5.1, where $m = \Omega(n)$ and Z_i is the i'th bit. Take any integer $0 < s < m$ and take any subset $S \subseteq [m]$ with $|S| = s$. Compute $W = \bigoplus Z_{i,i\in S}$. Then*

$$|W - U| = 2^{-\Omega(s)}.$$

Remark 5.10. Note that if we take $s = \Omega(n)$ then we get 1 bit that is $2^{-\Omega(n)}$-close to uniform.

Proof. Without loss of generality assume $S = \{1, \cdots, s\}$. The proof for the other cases are essentially the same. As in the proof of Theorem 5.2, we have Lemma 5.3, Lemma 5.4 and Lemma 5.5. Now consider $(U, V)_g, \cdots, (U, V)_t$ conditioned on the fixings of $(X_i = x_i, SR_i = sr_i, R_i = r_i, U_i = u_i)_{i \in \{1, \cdots, g-1\}}$ and $X_g = x_g, SR_g = sr_g$. Let $W_g = \bigoplus_{i=1}^{s} Z'_i$, where $Z'_i = \bigoplus_{j=g}^{t} V_{ji}$. Note that

$$W_g = \bigoplus_{i=1}^{s} Z'_i = \bigoplus_{i=1}^{s} (V_{gi} \oplus \bigoplus_{j=g+1}^{t} V_{ji}) = \bigoplus_{i=1}^{s} V_{gi} \oplus \bigoplus_{i=1}^{s} (\bigoplus_{j=g+1}^{t} V_{ji}).$$

We know that U_g is uniform and each V_{gi} is a degree c_g monomial on c_g bits of U_g. We also know that $P_g = \bigoplus_{i=1}^{s} (\bigoplus_{j=g+1}^{t} V_{ji})$ is a degree at most $d = c_g - 1$ polynomial of the bits of U_g. Let P_d stand for the class of all polynomials of degree at most $d = c_g - 1$ over $\mathsf{GF}(2)$. Then any polynomial in P_d can equal V_{gi} with probability at most $1 - 2^{-c_g} = 1 - 2^{-(d+1)}$. In other words,

$$\mathsf{Cor}(V_{gi}, P_d) \leq 1 - 1/2^d.$$

Note that V_{gi}'s are the same functions (degree c_g monomial) on different and thus independent bits of U_g (since U_g is uniform). The XOR lemma of Theorem 5.8 thus gives that

$$\mathsf{Cor}(\bigoplus_{i=1}^{s} V_{gi}, P_d) \leq 2^{-\Omega(s)}.$$

In particular, we have

$$\mathsf{Cor}(\bigoplus_{i=1}^{s} V_{gi}, \bigoplus_{i=1}^{s} (\bigoplus_{j=g+1}^{t} V_{ji})) \leq 2^{-\Omega(s)}.$$

Thus W_g is $2^{-\Omega(s)}$-close to uniform. Now by Lemma 5.3, Lemma 5.4 and Lemma 5.5, the error of $W = \bigoplus_{i=1}^{s} Z_i$ can go up by at most $2^{-\Omega(n)}$. Thus

$$|W - U| = 2^{-\Omega(n)} + 2^{-\Omega(s)} = 2^{-\Omega(s)}$$

since $s = O(n)$. ∎

We also need the following definition about asymptotically good binary linear codes:

Definition 5.11. A linear binary code of length n and rank k is a linear subspace C with dimension k of the vector space $\mathbb{F}_2^n$. If the distance of the code C is d we say that C is an $[n, k, d]_2$ code. C is asymptotically good if there exist constants $0 < \delta_1, \delta_2 < 1$ s.t. $k \geq \delta_1 n$ and $d \geq \delta_2 n$.

Note that every linear binary code has an associated generating matrix $G \in \mathbb{F}_2^{k \times n}$, and every codeword can be expressed as vG, for some vector $v \in \mathbb{F}_2^k$.

It is well known that we have explicit constructions of asymptotically good binary linear code. For example, the Justensen codes constructed in [Jus72].

Now we have the following construction and theorem:

Algorithm 5.12 ($\mathsf{AExt}(x)$)**.**

Input: x — an n bit string.
Output: z — an m bit string with $m = \Omega(n)$.

Sub-Routines and Parameters:
Let $\mathsf{ADisp} : \{0,1\}^n \to \{0,1\}^{m_1}$ be the affine disperser in Algorithm 5.1, where $m_1 = \Omega(n)$.
Let G be the generating matrix of an asymptotically good linear binary code with codeword length m_1. Thus G is an $\alpha m_1 \times m_1$ matrix for some constant $\alpha > 0$. Let G_i stand for the i'th row of the matrix.

1. Run Algorithm 5.1 and let the output be $Z = (Z_1, \cdots, Z_{m_1})$ where Z_i is the i'th bit.
2. For each codeword G_i, let $S_i = \{j \in [m_1] : G_{ij} = 1\}$ be the set of index s.t. the bit of the codeword G_i at that index is 1. Define
$$W_i = \bigoplus Z_{j,j\in S_i}$$
to be the bit associated with G_i, i.e., W_i is the XOR of the Z_j's whenever the j'th index of the codeword G_i is 1.
3. Take a constant $0 < \beta \leq \alpha$ to be chosen later. Output $W = (W_1, \cdots, W_{\beta m_1})$.

Theorem 5.13. *For every $\delta > 0$ there exists an efficient family of functions $\mathsf{AExt} : \{0,1\}^n \to \{0,1\}^m$ such that $m = \Omega(n)$ and for every affine source X with entropy δn, $|\mathsf{AExt}(X) - U_m| = 2^{-\Omega(n)}$.*

Proof. We show that Algorithm 5.12 is such a family of functions. First note that for any nonempty set $T \subseteq [\alpha m_1]$, the sum of the codewords $H_{i,i\in T}$ is also a codeword C_T. Let W_T be the bit associated with C_T, i.e., W_T is the XOR of the Z_j's whenever the j'th index of the codeword C_T is 1. Observe that
$$W_T = \bigoplus W_{i,i\in T}.$$
Since T is nonempty, the codeword C_T must have distance at least γm_1 from the codeword 0, for some constant $0 < \gamma < 1$. That is, C_T must have at least γm_1 1's. Thus by Theorem 5.9 $|W_T - U| = 2^{-\Omega(\gamma m_1)}$. This implies that for every nonempty subset T,
$$\left|\bigoplus W_{i,i\in T} - U\right| \leq 2^{-c_0\gamma m_1}$$
for some constant $c_0 > 0$. In other words, the random variables $\{W_i\}$ form an ϵ-biased space for $\epsilon = 2^{-c_0\gamma m_1}$. Thus by Lemma 2.24
$$|W - U| \leq 2^{\beta m_1/2} \cdot 2^{-c_0\gamma m_1}.$$
Choose $0 < \beta \leq \alpha$ s.t. $\beta \leq c_0\gamma$. Then

$$|W - U| \leq 2^{-c_0 \gamma m_1/2}.$$

Thus we have that $(W_1, \cdots, W_{\beta m_1})$ are $\Omega(n)$ bits that are $2^{-\Omega(n)}$-close to uniform. ■

6 Affine Extractors and Dispersers for Sub-Linear Entropy Sources

In this section we briefly show how we can modify the affine extractors and dispersers above to handle sources with slightly sub-linear entropy. The main observation is that in the construction of affine extractors for linear entropy, the polynomials that we use only have constant degrees. For an argument like the analysis of the extractor to hold, the degree of the polynomial can be as large as $\log n$ by Theorem 5.8. For an argument like the analysis of the disperser to hold, the degree of the polynomial can be close to n (the degree cannot be larger than n since we can get at most n uniform random bits). We'll show that this will lead to an affine extractor for entropy $n/\sqrt{\log \log n}$ and an affine disperser for entropy $n/\sqrt{\log n}$.

Theorem 6.1. *There exists a constant $c > 1$ and an efficient family of functions $\mathsf{ADisp} : \{0,1\}^n \to \{0,1\}^m$ such that $m = n^{\Omega(1)}$ and for every affine source X with entropy $cn/\sqrt{\log n}$, $|\mathsf{Supp}(\mathsf{ADisp}(X))| = 2^m$.*

Proof Sketch. We essentially use the same algorithm as Algorithm 5.1, except the entropy rate δ now is sub-constant. We examine the analysis to see how small δ can be. We focus on the first good block X_g.

First we want to use the somewhere condenser from Theorem 2.17 to convert a rate-$\delta/4$ source into a somewhere rate-$(1-\delta/4)$ source. Note that now δ is sub-constant, so we do this in two steps. First, we repeatedly use Theorem 2.16 to convert the source into a somewhere rate-0.6 source. This will take $O(\log \frac{1}{\delta})$ times. Next, we repeatedly use Theorem 2.20 to convert the source into a somewhere rate-$(1-\delta/4)$ source. This will take $O(\frac{1}{\delta})$ times. Thus in step 1 of Algorithm 5.1 we get that $\ell_1 = 2^{O(\frac{1}{\delta})}$, and each Y_{gj} has $n/(2^{O(\frac{1}{\delta})})$ bits. The error is $2^{-n/2^{O(\frac{1}{\delta})}}$.

Now in step 2 of Algorithm 5.1, we get $\ell_2 = 2^{O(\frac{1}{\delta})}$. Thus the total number of rows in the matrix SR_g is $\ell_1\ell_2 = 2^{O(\frac{1}{\delta})}$, with each row having $\delta^3 n/(3000\ell_1\ell_2) = n/(2^{O(\frac{1}{\delta})})$ bits. By Theorem 2.22 the error is $2^{-n/2^{O(\frac{1}{\delta})}}$.

In step 3, we apply $\mathsf{AffineSRExt}$. By Theorem 4.5 we get each R_g has $n/(2^{O(\frac{1}{\delta^2})})$ bits, with error $2^{-n/2^{O(\frac{1}{\delta^2})}}$. By Theorem 3.1 after applying LSExt, U_g has $n/(2^{O(\frac{1}{\delta^2})})$ bits with error $2^{-n/2^{O(\frac{1}{\delta^2})}}$.

The last thing to verify is that the degrees of the polynomials produced in step 5 satisfy the requirements as in the analysis of Theorem 5.2. The analysis says that we need to have $c_i > c(\delta)c_{i+1}$ for all i. To see how this can be satisfied, we first estimate the quantity $c(\delta)$.

As in the analysis of Theorem 5.2, first X is an affine function of U_g. Now by Theorem 2.16 and Theorem 2.20, each bit of $\mathrm{Zuc}(X_i)$ is a degree $2^{O(\frac{1}{\delta})}$ polynomial of the input bits (since we repeat the condenser $O(\frac{1}{\delta})$ times). The function Had is a degree 2 polynomial. Thus each bit of SR_i is a degree 2 polynomial of the input bits. Now we apply $\mathsf{AffineSRExt}$, and by Theorem 4.5 each bit of the output is a degree $2^{O(\frac{1}{\delta})}$ polynomial of the input bits. Therefore each bit of R_i is a degree $2^{O(\frac{1}{\delta})} \cdot 2^{O(\frac{1}{\delta})} = 2^{O(\frac{1}{\delta})}$ polynomial of the bits of U_g. By Theorem 3.1 in LSExt each bit of U_i is a constant degree polynomial of the input bits. Thus we conclude that for every $i \geq g+1$, each bit of U_i is a degree $2^{O(\frac{1}{\delta})}$ polynomial of the bits of U_g.

Therefore we have

$$c(\delta) = 2^{O(\frac{1}{\delta})}.$$

Note that we need $c_i > c(\delta)c_{i+1}$ for every $i, 1 \leq i \leq 10/\delta$. Thus the c_i's are bounded by

$$c(\delta)^{10/\delta} = 2^{O(\frac{1}{\delta^2})}.$$

Since each U_i has $n/(2^{O(\frac{1}{\delta^2})})$ bits, it suffices to have

$$n/(2^{O(\frac{1}{\delta^2})}) > 2^{O(\frac{1}{\delta^2})}.$$

Thus by taking $\delta \geq c/\sqrt{\log n}$ for some constant $c > 1$ the disperser can output $n^{\Omega(1)}$ bits. ■

Similarly, we get an affine extractor for sub-linear entropy sources. However, unlike in the analysis of the affine disperser, the degree of the polynomial cannot be close to n, and can only be close to $\log n$ by Theorem 5.8. Thus we only get $\delta = c/\sqrt{\log \log n}$ for some constant $c > 1$.

Theorem 6.2. *There exists a constant $c > 1$ and an efficient family of functions $\mathsf{AExt} : \{0,1\}^n \to \{0,1\}^m$ such that $m = n^{\Omega(1)}$ and for every affine source X with entropy $cn/\sqrt{\log \log n}$, $|\mathsf{AExt}(X) - U_m| = 2^{-n^{\Omega(1)}}$.*

Proof Sketch. We essentially use the same algorithm as Algorithm 5.12, except the entropy rate δ now is sub-constant. We examine the analysis to see how small δ can be. We focus on the first good block X_g.

As in the analysis of the affine disperser above, we get that in step 4 of Algorithm 5.1, U_g has $n/(2^{O(\frac{1}{\delta^2})})$ bits, and the error is $2^{-n/2^{O(\frac{1}{\delta^2})}}$. We also get that

$$c(\delta) = 2^{O(\frac{1}{\delta})}$$

and thus the c_i's are bounded by

$$c(\delta)^{10/\delta} = 2^{O(\frac{1}{\delta^2})}.$$

For the xor lemma of Theorem 5.8 to give a non-trivial bound, it suffices to have

$$2^{O(\frac{1}{\delta^2})} \leq \alpha \log n$$

for some constant $0 < \alpha < 1$. This gives that $\delta \geq c/\sqrt{\log \log n}$ for some constant $c > 1$. Also, when this happens, the XOR lemma of Theorem 5.8 gives a correlation upper bound of $2^{-n^{\Omega(1)}}$. The error of U_g is $2^{-\Omega(n/\log n)}$. Thus by taking the generating matrix of a binary linear asymptotically good code and choosing $n^{\Omega(1)}$ rows, we see that the extractor outputs $n^{\Omega(1)}$ bits that are $2^{-n^{\Omega(1)}}$-close to uniform. ■

Acknowledgements

We thank David Zuckerman and Yael Tauman Kalai for helpful discussions.

References

[BKS+05] Boaz Barak, Guy Kindler, Ronen Shaltiel, Benny Sudakov, and Avi Wigderson. Simulating independence: New constructions of condensers, Ramsey graphs, dispersers, and extractors. In *Proc. of 37th STOC*, pages 1–10, 2005.

[BSK09] Eli Ben-Sasson and Swastik Kopparty. Affine dispersers from subspace polynomials. In *Proceedings of the 41th Annual ACM Symposium on Theory of Computing*, 2009.

[BKS+09] Arnab Bhattacharyya, Swastik Kopparty, Grant Schoenebeck, Madhu Sudan, and David Zuckerman. Optimal testing of reed-muller codes. *Electronic Colloquium on Computational Complexity (ECCC)*, 2009.

[Bou07] Jean Bourgain. On the construction of affine-source extractors. *Geometric and Functional Analysis*, 1:33–57, 2007.

[CG88] Benny Chor and Oded Goldreich. Unbiased bits from sources of weak randomness and probabilistic communication complexity. *SIAM Journal on Computing*, 17(2):230–261, 1988.

[DG10] Matt DeVos and Ariel Gabizon. Simple affine extractors using dimension expansion. In *Proceedings of the 25th Annual IEEE Conference on Computational Complexity*, 2010.

[DKSS09] Zeev Dvir, Swastik Kopparty, Shubhangi Saraf, and Madhu Sudan. Extensions to the method of multiplicities, with applications to kakeya sets and mergers. In *Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science*, pages 181–190, 2009.

[DW08] Zeev Dvir and Avi Wigderson. Kakeya sets, new mergers and old extractors. In *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science*, 2008.

[FS02] Lance Fortnow and Ronen Shaltiel. Recent developments in explicit constructions of extractors, 2002.

[GR05] Ariel Gabizon and Ran Raz. Deterministic extractors for affine sources over large fields. In *Proc. of 46th FOCS*, 2005.

[GRS04] Ariel Gabizon, Ran Raz, and Ronen Shaltiel. Deterministic extractors for bit-fixing sources by obtaining an independent seed. In *Proc. of 45th FOCS*, 2004.

[Gol95] Oded Goldreich. Three xor-lemmas - an exposition. *Electronic Colloquium on Computational Complexity (ECCC)*, 2(56), 1995.

[ILL89] R. Impagliazzo, L. Levin, and M. Luby. Pseudo-random generation from one-way functions. In *Proc. of 21st STOC*, pages 12–24, 1989.

[Jus72] J. Justensen. A class of constructive asymptotically good algebraic codes. *IEEE Trans. Info. Theory.*, 18:652656, 1972.

[KLR09] Yael Kalai, Xin Li, and Anup Rao. 2-source extractors under computational assumptions and cryptography with defective randomness. In *Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science*, 2009.

[KRVZ06] Jesse Kamp, Anup Rao, Salil Vadhan, and David Zuckerman. Deterministic extractors for small space sources. In *Proc. of 38th STOC*, 2006.

[KZ07] Jesse Kamp and David Zuckerman. Deterministic extractors for bit-fixing sources and exposure-resilient cryptography. *SIAM Journal on Computing*, 36(5):1231–1247, 2007.

[LRVW03] C. J. Lu, Omer Reingold, Salil Vadhan, and Avi Wigderson. Extractors: Optimal up to constant factors. In *Proc. of 35th STOC*, pages 602–611, 2003.

[Rao09] Anup Rao. Extractors for low-weight affine sources. In *Proceedings of the 24nd Annual IEEE Conference on Computational Complexity*, 2009.

[Tre01] Luca Trevisan. Extractors and pseudorandom generators. *Journal of the ACM*, pages 860–879, 2001.

[TV00] Luca Trevisan and Salil Vadhan. Extracting randomness from samplable distributions. In *Proc. of 41st FOCS*, pages 32–42, 2000.

[Vaz85] Umesh Vazirani. Towards a strong communication complexity theory or generating quasi-random sequences from two communicating slightly-random sources (extended abstract). In *Proceedings of the 17th Annual ACM Symposium on Theory of Computing*, pages 366–378, 1985.

[Vin07] Le Anh Vinh. Szemeredi-trotter type theorem and sum-product estimate in finite fields. In *Arxiv*, 2007.

[VW08] Emanuele Viola and Avi Wigderson. Norms, xor lemmas, and lower bounds for polynomials and protocols. *Theory of Computing*, 4(7):137–168, 2008.

[Yeh10] Amir Yehudayoff. Affine extractors over prime fields. *Manuscript*, 2010.

[Zuc07] David Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. In *Theory of Computing*, pages 103–128, 2007.

ECCC ISSN 1433-8092

http://eccc.hpi-web.de