Electronic Colloquium on Computational Complexity, Revision 2 of Report No. 155 (2013)

Pseudorandom Generators for Low Degree Polynomials from Algebraic Geometry Codes

Gil Cohen[*] Amnon Ta-Shma[†]

March 12, 2014

Abstract

Constructing pseudorandom generators for low degree polynomials has received a considerable attention in the past decade. Viola [CC 2009], following an exciting line of research, constructed a pseudorandom generator for degree d polynomials in n variables, over any prime field. The seed length used is $O(d \log n + d2^d)$, and thus this construction yields a non-trivial result only for $d = O(\log n)$. Bogdanov [STOC 2005] presented a pseudorandom generator with seed length $O(d^4 \log n)$. However, it is promised to work only for fields of size $\Omega(d^{10} \log^2 n)$. The work of Lu [CCC 2012], combined with that of Bogdanov, yields a pseudorandom generator with seed length $O(d^4 \log n)$ for fields of size $\Omega(d^{6+c})$ – independent of n, where c is an arbitrarily small constant. Based on these works, Guruswami and Xing [CCC 2014] devised a construction with a similar seed length for fields of size $O(d^6)$.

In this work we show that for any d, a random sub-code (with a proper dimension) of any good algebraic geometry code, is a hitting set for degree d polynomials. By derandomizing this assertion, together with the work of Bogdanov, we obtain a construction of a pseudorandom generator for degree d polynomials over fields of size $O(d^{12})$, and seed length $O(d^4 \log n)$. The running-time of our construction is $n^{\mathrm{poly}(d)}$. However, the running-time can be improved to $\mathrm{poly}(n, d)$ assuming Riemann-Roch spaces of certain algebraic function fields are, in some sense, strongly explicit. We believe this open problem is interesting on its own, and take a first step at affirming the conjecture.

Although quantitatively our result does not match the parameters of Guruswami and Xing, our construction is clean mathematically and conceptually simpler. We consider the proof technique to be the main contribution of this paper, and believe it will find other applications in complexity theory. In the heart of our proofs is a reduction from the problem of assuring independence between monomials to the much simpler problem of avoiding collisions over the integers. Our reduction heavily relies on the Riemann-Roch theorem.

[*]Department of Computer Science and Applied Mathematics, Weizmann Institute of Science, Rehovot 76100, Israel. `Email: gil.cohen@weizmann.ac.il`. Supported by an ISF grant and by the I-CORE Program of the Planning and Budgeting Committee.

[†]The Blavatnik School of Computer Science, Tel-Aviv University, Israel, 69978. `Email: amnon@cs.tau.ac.il`. Supported by ISF grant no. 1090/10.

ISSN 1433-8092

1 Introduction

A *pseudorandom generator* for degree d polynomials is a function $G \colon \{0,1\}^\ell \to \mathbb{F}^n$ that "fools" any degree d polynomial in n variables over a field $\mathbb{F}$. [1] That is, a random output of G is ε-close, in statistical distance, to a uniformly random element of $\mathbb{F}^n$, from the point of view of any degree d polynomial. The input for G is called the seed. Naturally, the goal is to devise an efficiently computable pseudorandom generator with seed length ℓ as small as possible in terms of n, d and ε^{-1}. In the introduction, for simplicity, we think of ε as some small fixed constant, say, $1/10$. A probabilistic argument shows that, computational aspects aside, there exists a pseudorandom generator for degree d polynomials with seed length $O(d \log n)$, and this upper bound is optimal [BV10].[2]

Small-bias sets, introduced by Naor and Naor [NN93], are pseudorandom generators for linear functions (that is, $d = 1$), and explicit constructions with optimal seed length $\ell = O(\log n)$ are known [NN93, ABN+92, AGHP92, BT09]. The first result to handle larger degrees was obtained by Luby, Veličković and Wigderson [LVW93] (simplified by Viola [Vio07]), which yields a pseudorandom generator for constant degree polynomials over $\mathbb{F}_2$ with non-trivial, yet far from optimal, seed length $2^{O(\sqrt{\log n})}$. Over the last decade, the problem of constructing pseudorandom generators for degree d polynomials, for any given d, has received considerable attention, and we briefly review this exciting line of research and the techniques that were involved. See Table 1 for a summary of these results.

The first breakthrough in this line of research was obtained by Bogdanov [Bog05]. By considering "large fields", namely, fields with size that is allowed to depend on n, d, Bogdanov [Bog05] gave a pseudorandom generator that fools degree d polynomials for all d, assuming $|\mathbb{F}| \geq \Omega(d^{10} \log^2 n)$, having seed length $O(d^4 \log n)$. Bogdanov applied classical results from algebraic geometry for his construction.

A key result in Bogdanov's work is a novel black-box reduction from the construction of pseudorandom generators for degree d polynomials to the construction of hitting set generators with density close to 1, for degree $\Omega(d^4)$ polynomials. A *hitting set generator* of density $1 - \delta$ for degree d polynomials in $\mathbb{F}[y_1, \ldots, y_n]$ is a function $H \colon \{0,1\}^\ell \to \mathbb{F}^n$ such that for every degree d polynomial $f \in \mathbb{F}[y_1, \ldots, y_n]$, it holds that $\mathbf{Pr}_{x \sim \{0,1\}^\ell}[f(H(x)) = 0] \leq \delta$. Bogdanov's reduction can only yield pseudorandom generators for fields of size $\Omega(d^6)$, regardless of the field size required by the hitting set generator. However, on the positive side, if the hitting set generator works for fields of size independent of n, then so does the resulting pseudorandom generator.

Lu [Lu12], among other results, gave a construction of a hitting set generator for low degree polynomials, with better parameters than those obtained by Bogdanov. The key idea in Lu's construction is to take *several* dependent samples of a small-bias set (over a finite field with an appropriate size), where the samples are taken by performing a random walk on an expander. Quantitatively, together with Bogdanov's reduction, Lu's work yields a pseudorandom generator for fields of size as small as $O(d^{6+c})$, using $O((d^4/c) \cdot \log n)$ random

[1]By "degree" we mean total degree.

[2]By "optimal" we mean optimal up to a multiplicative constant factor.

bits, for any desired $c \in (0,1)$. [3] Very recently (in fact, after the publication of this paper) Guruswami and Xing [GX13] devised a pseudorandom generator for fields with size $O(d^6)$ that uses $O(d^4 \log n)$ random bits.

Going back to constant size fields (namely, fields with size independent of n, d; in particular $\mathbb{F}_2$), Bogdanov and Viola [BV10] completely strayed from the work [Bog05], and suggested the elegant claim stating that the sum of d independent samples of any small-bias set fools degree d polynomials. Bogdanov and Viola proved this claim assuming the d vs. $d-1$ inverse conjecture for the Gowers norm. The resulting seed length is $O(d \log n) + f(d, |\mathbb{F}|)$, where f is some function of $d, |\mathbb{F}|$. Bogdanov and Viola gave an unconditional proof for the case of quadratic and cubic polynomials (i.e., $d = 2, 3$) over any prime field. Not too long afterwards, the above conjecture was proved for $|\mathbb{F}| > d$ by Green and Tao [GT07]. Together, these two results imply a pseudorandom generator for any degree d polynomial, with seed length as above, assuming $|\mathbb{F}| > d$, .

Somewhat surprisingly, the d vs. $d-1$ inverse conjecture for the Gowers norm was proved to be false for $|\mathbb{F}| \le d$, as discovered independently by Green and Tao [GT07] and by Lovett, Meshulam and Samorodnitsky [LMS08]. Nevertheless, Lovett [Lov08] proved, unconditionally, that the sum of 2^d independent samples from any small-bias set fools degree d polynomials, over any prime field. Moreover, Kaufman and Lovett [KL08] observed that the proof of Bogdanov and Viola can be conditioned on a weaker claim than the (false) d vs. $d-1$ inverse conjecture for the Gowers norm, and this weaker claim was proved in [KL08] for all prime fields.

Finally, Viola [Vio09b] gave a surprisingly simple proof, based on basic Fourier analysis, for the claim that over any prime field, the sum of d independent samples from a small-bias set fools degree d polynomials. Viola's analysis improved upon the results mentioned above in all aspects (i.e., seed length and field size), excluding Lu's pseudorandom generator, which has incomparable parameters. The only remaining caveat in Viola's construction is that the seed length has an exponential dependence on d (more precisely, it is $O(d \log n + d \cdot 2^d)$). Thus, it gives no meaningful pseudorandom generator for degree $d = \Omega(\log n)$ (see also [Vio09a]).

1.1 Our Results

In this work we give a clean mathematical construction of a hitting set generator with density close to 1 for low degree polynomials, based on the deep and fundamental Riemann-Roch theorem from algebraic geometry. By applying Bogdanov's reduction we obtain a pseudorandom generator.

In Section 1.2 we give a detailed, though informal, overview of our proofs. However, we consider the proof technique to be the main contribution of this paper, and so, we start this section by giving a very informal description of our proof technique.

Consider a linear code $C \subseteq \mathbb{F}_q^m$ with dimension n and relative distance $1-\delta$. The code C can be viewed as a hitting set generator with density $1-\delta$ for degree 1 polynomials over $\mathbb{F}_q[y_1, \ldots, y_n]$ in the following way: Let $c_1, \ldots, c_n$ be some basis for C. The hitting set

[3]This is somewhat implicit in Lu's work (see the paragraph following Theorem 1 in [Lu12]).

generator induced by C is defined by sampling a coordinate $r \sim [m]$ uniformly at random and outputting the sequence $((c_1)_r, \dots, (c_n)_r) \in \mathbb{F}_q^n$. To see that this is indeed a hitting set generator with density $1-\delta$, consider a non-zero linear function $f(y_1, \dots, y_n) = \alpha_1 y_1 + \cdots + \alpha_n y_n$ over $\mathbb{F}_q$. By applying f on a random output of the above (claimed to be) hitting set generator, one gets the field element $\alpha_1(c_1)_r + \cdots + \alpha_n(c_n)_r = (\alpha_1 c_1 + \cdots + \alpha_n c_n)_r$. Since not all α_i's are zeros, $\alpha_1 c_1 + \cdots + \alpha_n c_n$ is a non-zero codeword in C, and thus, the probability over r that $(\alpha_1 c_1 + \cdots + \alpha_n c_n)_r = 0$ is at most δ, as desired. By considering optimal algebraic geometry codes, one can take m and q to be as small as $m = O(n/\delta)$ and $q = O(1/\delta^2)$.

Clearly, the idea above breaks even for degree 2 polynomials [4] – after all, what is the meaning of *multiplying* two codewords? However, there is a meaning for multiplying codewords of an algebraic geometry code (and of any evaluation code in general), and our approach for constructing a hitting set generator for low degree polynomials exploits this extra structure (see Section 1.2). In particular, we prove that for any d, with high probability, a random sub-code with an appropriate dimension of an algebraic geometry code is a hitting set generator for degree d polynomials.

The claim mentioned above, which is implicit in our proofs, relies on the Riemann-Roch theorem. At the heart of the proof lies a reduction from the problem of assuring independence between the monomials of the polynomial we are trying to fool, to that of avoiding collisions over the integers. We perform a suitable derandomization of the above statement, and together with Bogdanov's reduction, obtain the following explicit construction of a pseudorandom generator.

Theorem 1.1. *Let p be a prime power, and let $q = p^2$. Let $d \in \mathbb{N}$ and $\varepsilon > 0$ be such that $q = \Omega(d^{12}/\varepsilon^4)$. Then, there exists a pseudorandom generator with bias ε for degree d polynomials in $\mathbb{F}_q[y_1, \dots, y_n]$, with seed length $O(d^4 \log(n) + \log(1/\varepsilon))$. The running-time of the pseudorandom generator is* $\mathrm{poly}(n^{d^4}, 1/\varepsilon)$. *Moreover, assuming Conjecture 1.4 (see Section 1.3) the running-time is* $\mathrm{poly}(n, d, \log(1/\varepsilon))$.

Although Theorem 1.1's field size and running time do not match the best known construction [GX13], we believe our construction of a hitting set generator has the advantage of being cleaner mathematically, and in particular it does not rely on any previous work. We consider the novel proof technique to be the main contribution of this paper, and believe it will find other applications. We also give an alternative construction that has running-time $\mathrm{poly}(n, d, \log(1/\varepsilon))$ and seed length $O(d^4 \log(n) + \log(1/\varepsilon))$. However, it is promised to work only for fields of size depending also on n.

Theorem 1.2. *Let p be a prime power, and let $q = p^2$. Let $n, d \in \mathbb{N}$ and $\varepsilon > 0$ be such that*

$$q = \Omega\left(\left(\frac{d^{10}}{\varepsilon^2} \cdot \frac{\log(n/\varepsilon)}{\log\log n}\right)^2\right).$$

Then, there exists a pseudorandom generator with bias ε for degree d polynomials in $\mathbb{F}_q[y_1, \dots, y_n]$, with seed length $O(d^4 \log(n) + \log(1/\varepsilon))$. The running-time of the pseudorandom generator is $\mathrm{poly}(n, d, \log(1/\varepsilon))$.

[4] In fact, it also breaks when the linear function has a constant coefficient $\alpha_0 \neq 0$.

1.2 Proof Overview

As mentioned above, by Bogdanov's reduction (see Theorem 2.3), to construct a pseudorandom generator for degree d polynomials, it is enough to construct a hitting set generator with density close to 1 for degree poly(d) polynomials. In order to describe our hitting set generator, we also consider the number of monomials a given polynomial has (i.e., its sparsity), which we denote by s. We do so as our proofs work naturally with respect to s. It should be noted that constructing pseudorandom generators (and hitting set generators) for sparse polynomials has also received attention in the literature (see, e.g., [KS01, Bog05, Lu12]).

In the following proof overview we make use of standard notations and results from algebraic function fields. The unfamiliarized reader is referred to Section 2 for the formal definitions.

Let $F/\mathbb{F}_q$ be an algebraic function field. Assume P_∞ is a rational place of $F/\mathbb{F}_q$. For an integer r, consider the Riemann-Roch space $\mathcal{L}(r \cdot P_\infty)$ with dimension n and basis vectors $f_1, \ldots, f_n$. We recall that the Riemann-Roch theorem implies that if $r \geq 2g(F) - 1$ then $n = r - g(F) + 1$. However, in this proof overview, we allow ourselves to assume for simplicity that $n = r$, and that $v_\infty(f_i) = -i$ for $i = 1, \ldots, n$. Even though such an assumption cannot be met, it would be illustrative to show our ideas.

Influenced by the novel construction of algebraic geometry codes, introduced by Goppa [Gop81] (see [Sti93], Chapter 2) and by the construction of small-bias sets based on algebraic geometry codes by Ben-Aroya and Ta-Shma [BT09], a first attempt for a construction of a hitting set generator for degree d polynomials is the following.

The failing hitting set generator

1. Sample a rational place $P \neq P_\infty$ uniformly at random.
2. Output $(f_1(P), \ldots, f_n(P))$.

As in Goppa's argument, we note that since $P \neq P_\infty$, each f_i can be evaluated at P and thus the output is well-defined. Moreover, the output is indeed a vector in $\mathbb{F}_q^n$ since P is rational.

To analyze this attempt of a hitting set generator, consider a non-zero degree d polynomial $h \in \mathbb{F}_q[y_1, \ldots, y_n]$ having s monomials. The hitting set generator above maps a basis function f_i, evaluated at a random rational place P, to each variable y_i. This induces a mapping from each monomial $y_1^{d_1} \cdots y_n^{d_n}$ of h to the function $f_1^{d_1} \cdots f_n^{d_n}$ (which is then evaluated at P). We note that the latter function is contained in $\mathcal{L}(nd \cdot P_\infty)$ as it has no poles outside P_∞, and

$$v_\infty(f_1^{d_1} \cdots f_n^{d_n}) = \sum_{i=1}^{n} d_i v_\infty(f_i) = \sum_{i=1}^{n} d_i(-i) \geq \sum_{i=1}^{n} d_i(-n) \geq -nd.$$

Thus, h is mapped to a function $f \in \mathcal{L}(nd \cdot P_\infty)$. If we knew f is a non-zero function then an argument similar to that of Goppa would complete the proof. However, it could be the case that the monomials of h induce functions that are linearly dependent over $\mathbb{F}_q$. This may result in h being mapped to $f = 0$.

To overcome this obstacle, we recall that functions with different valuations are linearly independent over $\mathbb{F}_q$. Thus, if we could guarantee that all monomials of h are mapped to functions with distinct valuations, then no cancelation can occur, and in particular, f would be a non-zero function. [5] We thus reduced the question of the linear independence of the induced functions to a question about avoiding collisions over the integers.

We stress that the fact that for every prescribed valuation $v \geq 2g(F) - 1$, there exists a function $f \in \mathcal{L}(v \cdot P_\infty) \setminus \mathcal{L}((v-1) \cdot P_\infty)$ is very deep and follows by the Riemann-Roch theorem. This is what enables us to carry out the reduction.

Going back to the technical details, to avoid the collisions over the integers we use randomness. The following is a hitting set generator that actually works, yet is extremely wasteful in terms of random bits.

The working, yet wasteful, hitting set generator

1. Sample $(Z_1, \ldots, Z_n)$ uniformly at random from $[m]^n$, where m is a parameter we fix later on.
2. For each $i \in [n]$, find a function $f_i \in \mathcal{L}(Z_i \cdot P_\infty) \setminus \mathcal{L}((Z_i - 1) \cdot P_\infty)$.
3. Sample a rational place $P \neq P_\infty$ uniformly at random.
4. Output $(f_1(P), \ldots, f_n(P))$.

To analyze this hitting set generator, consider two distinct monomials $y_1^{d_1} \cdots y_n^{d_n}$, $y_1^{d'_1} \cdots y_n^{d'_n}$ of h. These monomials are mapped to the functions $f_1^{d_1} \cdots f_n^{d_n}$, $f_1^{d'_1} \cdots f_n^{d'_n}$ respectively. Note that $v_\infty(f_1^{d_1} \cdots f_n^{d_n}) = \sum_{i=1}^n d_i Z_i$ and $v_\infty(f_1^{d'_1} \cdots f_n^{d'_n}) = \sum_{i=1}^n d'_i Z_i$. Thus, the valuation of these two functions will collide if and only if

$$\sum_{i=1}^{n} (d_i - d'_i) Z_i = 0. \tag{1}$$

Since the Z_i's are chosen uniformly and independently at random from $[m]$, Equation (1) holds with probability at most $1/m$. Thus, by union bound over all pairs of h's monomials, except with probability $\binom{s}{2} \cdot \frac{1}{m}$, the monomials of h are mapped to functions with distinct valuations. By taking $m = s^2/\delta$, we have that $f \in \mathcal{L}(md \cdot P_\infty)$ is a non-zero function, except with probability δ. Conditioned on this event, one can follow Goppa's argument to show that the algorithm above is indeed a hitting set generator with density close to 1.

Saving on random bits. Although the hitting set generator above works, it requires $\Omega(n \log s)$ random bits (while $O(n \log |\mathbb{F}|)$ random bits are suffice to sample a uniform n tuple over $\mathbb{F}$). One way to save on random bits is to exploit the locality of the "test"

[5] In fact, it is enough to guarantee that the minimal valuation is obtained by exactly one function. We believe that this fact can be used to slightly improve the sample space size of our construction.

for the Z_i's in Equation (1). Indeed, at most $2d$ of the n summands are non-zero. Taking advantage of this locality can be done by using a $2d$-wise independent family of hash functions $\mathcal{H} = \{h\colon [n] \to [m]\}$, where for simplicity we take δ to be constant and so $m = O(s^2)$. More precisely, Z_i is obtained by sampling $h \sim \mathcal{H}$ and setting $Z_i = h(i)$. This idea can be shown to work, and the number of random bits used is substantially lower – $O(d \log(ns))$, which is $O(d^2 \log n)$ under no assumption on the sparsity (i.e., $s = O(n^d)$, which follows by the bound on the degree).

In our constructions we sample the sequence $(Z_1, \ldots, Z_n)$ using even fewer random bits – $O(\log(sd \log n))$, by exploiting the fact that the test for the randomness of the Z_i's in Equation (1) is not only local but in fact a sparse linear function with bounded coefficients (see Section 3). Under no assumption on the sparsity, this boils down to $O(d \log n)$ random bits. We call these "supporting" pseudorandom object which we construct *ε-biased for linear tests modulo m with d-bounded coefficients.*

Computing a function with a given valuation. Another issue we point out is that the algorithm assumes it can find a function $f \in \mathcal{L}(r \cdot P_\infty) \setminus \mathcal{L}((r-1) \cdot P_\infty)$, given r. Finding a basis for $\mathcal{L}(r \cdot P_\infty)$ can be done in time $\mathrm{poly}(r)$, when F is the Garcia-Stichtenoth tower [SAK+01] (and more generally using Heß's algorithm [Heß02]). Thus, one can find f in time $\mathrm{poly}(r)$. However, r can be as large as $\Omega(n^d)$. We conjecture that finding f can be done in time $\mathrm{polylog}(r)$ (see Section 1.3). Unconditionally, we show how to find f as above in time $\mathrm{polylog}(r)$ for large enough r (see Theorem 1.5). This allows us to construct a hitting set generator that runs in time $\mathrm{poly}(n)$, as apposed to $n^{\mathrm{poly}(d)}$. However, this introduces a dependency of the field size in n, s (see Theorem 1.2).

1.3 The Explicitness of $\mathcal{L}(r \cdot P_\infty)$ in the Garcia-Stichtenoth Tower of Function Fields

Definition 1.3. *Let $\mathcal{F} = \{F_k/\mathbb{F}_q\}_{k=0}^{\infty}$ be a family of function fields. Let*

$$G = \{G^{(k)} \mid G^{(k)} \text{ is a rational place of } F_k\}.$$

We say $\{\mathcal{L}(r \cdot G^{(k)})\}_{r,k}$ is fully explicit *if there exists an algorithm that on input k, r such that $r \geq 2g(F_k) - 1$, finds a function $f \in \mathcal{L}(r \cdot G^{(k)}) \setminus \mathcal{L}((r-1) \cdot G^{(k)})$ in time* $\mathrm{polylog}(r)$.

For concreteness we work with the Garcia-Stichtenoth tower of function fields over $\mathbb{F}_q$, where p is a prime power and $q = p^2$ [GS96] (see Preliminaries). We consider $G^{(k)} = P_\infty^{(k)}$. Our conjecture states that $\{\mathcal{L}(r \cdot P_\infty^{(k)})\}_{r,k}$ is fully explicit. Namely,

Conjecture 1.4. *Let p be a prime power, and let $q = p^2$. Consider the k^{th} level, F_k, of the Garcia-Stichtenoth tower. Then, for any integer $r \geq 2g(F_k) - 1$, one can find a function $f \in \mathcal{L}(r \cdot P_\infty^{(k)}) \setminus \mathcal{L}((r-1) \cdot P_\infty^{(k)})$ in time* $\mathrm{polylog}(r)$.

Shum *et al.* [SAK+01] devised an algorithm that given k, r as inputs, such that $r \geq 2g(F_k)-1$, runs in time $\mathrm{poly}(r)$ and computes a basis for $\mathcal{L}(r \cdot P_\infty^{(k)})$. Therefore, one can find some function $f \in \mathcal{L}(r \cdot P_\infty^{(k)}) \setminus \mathcal{L}((r-1) \cdot P_\infty^{(k)})$ in time $\mathrm{poly}(r)$. The algorithm of Shum *et al.* seems to be inherently "global", in the sense that it heavily relies on finding all basis vectors in order to find f. Heß's algorithm [Heß02] also seems to rely on a global approach. Conjecture 1.4 asserts that a function with a given valuation r at $P_\infty^{(k)}$ (having no poles elsewhere), can be found "locally", without resorting to the computation of all basis vectors. In fact, even an $\exp(\sqrt[5]{\log r})$-time algorithm suffices for the pseudorandom generator from Theorem 1.1 to run in time $\mathrm{poly}(n)$, for $d < \log n$.

It is worth noting that $\{\mathcal{L}(r \cdot P_\infty^{(k)})\}_{r,k}$ in the Hermitian tower of function fields is fully explicit. Indeed, a basis for $\mathcal{L}(r \cdot P_\infty^{(k)})$ in this tower is given by

$$\left\{ x_1^{i_1} \cdots x_k^{i_k} \;\middle|\; i_1, \ldots, i_k \geq 0, \quad i_2, \ldots, i_k \leq p-1 \text{ and } \sum_{j=1}^{k} i_j p^{k-j}(p+1)^{j-1} \leq r \right\}.$$

However, the algebraic geometry code that is based on this tower is not a good code. One can instantiate our meta pseudorandom generator with this tower, however in Theorem 1.2 we obtain better parameters.

We make a first step at affirming Conjecture 1.4 by proving that the conjecture does hold for many values of r.

Theorem 1.5. *Let p be a prime power, and let $q = p^2$. There exists an algorithm that given integers r, k, p such that*

$$kp^{k+1} - \frac{p^k - 1}{p-1} - p^k + 1 \leq r \leq kp^{k+1} - \frac{p^k - 1}{p-1},$$

finds a function $f \in \mathcal{L}(r \cdot P_\infty^{(k)}) \setminus \mathcal{L}((r-1) \cdot P_\infty^{(k)})$ in the Garcia-Stichtenoth tower over $\mathbb{F}_q$. The running time of the algorithm is $\mathrm{polylog}(r)$.

This weaker version of the conjecture enables us to prove Theorem 1.2. The proof of Theorem 1.5 follows straightforwardly from the work of Pellikaan, Stichtenoth and Torres [PST98], and we give it in Section 5.1.

2 Preliminaries

We denote by $\log(\cdot)$ the logarithm to base 2. The set $\{1, \ldots, n\}$ is denoted by $[n]$. Throughout the paper, for readability, we suppress flooring and ceiling.

Pseudorandom generators and hitting set generators for low degree polynomials

Definition 2.1. *A function $G\colon \{0,1\}^\ell \to \mathbb{F}^n$ is called a* pseudorandom generator *with bias ε for degree d polynomials, if for every degree d polynomial $f \in \mathbb{F}[y_1, \ldots, y_n]$, the statistical distance between $f(x)$ and $f(G(y))$, where x, y are sampled uniformly from $\mathbb{F}^n$ and $\{0,1\}^\ell$ respectively, is at most ε.*

Definition 2.2. *A function $H\colon \{0,1\}^\ell \to \mathbb{F}^n$ is called a* hitting set generator *of density $1-\delta$ for degree d polynomials, if for every non-zero degree d polynomial $f \in \mathbb{F}[y_1,\dots,y_n]$, $\mathbf{Pr}_{x\sim\{0,1\}^\ell}[f(H(x))=0] \le \delta$.*

Bogdanov [Bog05] gave a black-box reduction from the construction of pseudorandom generators for low degree polynomials to the construction of hitting set generators with density close to 1 for low degree polynomials.

Theorem 2.3 ([Bog05], Theorem 3.1 restated)**.** *Let $G_1\colon \{0,1\}^{\ell_1} \to \mathbb{F}_q^{2n-1}$ be a hitting set generator of density $1-\delta$ for polynomials of degree $3d^2$. Let $G_2\colon \{0,1\}^{\ell_2} \to \mathbb{F}_q^{n-1}$ be a hitting set generator of density $1-\delta$ for polynomials of degree $3d^4$. Suppose that G_1 maps a seed x_1 to $(v_1,\dots,v_n,w_2,\dots,w_n) \in \mathbb{F}_q^{2n-1}$ and G_2 maps a seed x_2 to $(z_2,\dots,z_n) \in \mathbb{F}_q^{n-1}$. Then, the map $G'\colon \{0,1\}^{\ell_1+\ell_2} \times \mathbb{F}_q^2 \to \mathbb{F}_q^n$ given by $G'(x_1,x_2,s,t) = (s+v_1, w_2s+z_2t+v_2,\dots,w_ns+z_nt+v_n)$ is a pseudorandom generator for degree d polynomials, with bias $O(\sqrt{\delta}d + d^2q^{-1/2} + d^6q^{-1})$.*

Background of Algebraic Function Fields

In this section we recall standard notions from the theory of algebraic function fields (see, e.g., [Sti93]). For a prime power q, we denote the field with q elements by $\mathbb{F}_q$. The *rational function field* is denoted by $\mathbb{F}_q(x)$, where x is some transcendental element over $\mathbb{F}_q$. An *algebraic function field* over $\mathbb{F}_q$, denoted by $F/\mathbb{F}_q$, is a finite algebraic extension of $\mathbb{F}_q(x)$. A *discrete valuation* of $F/\mathbb{F}_q$ is a function $v\colon F \to \mathbb{Z} \cup \{\infty\}$ with the following properties:

1. $v(f) = 0$ for all non-zero $f \in \mathbb{F}_q$.
2. $v(f) = \infty$ if and only if $f = 0$.
3. $v(fg) = v(f) + v(g)$ for all $f, g \in F$.
4. $v(f+g) \ge \min(v(f), v(g))$.
5. There exists $f \in F$ such that $v(f) = 1$.

In fact, it follows that for $f, g \in F$ with distinct valuations, $v(f+g) = \min(v(f), v(g))$. In particular, elements $f_1,\dots,f_s \in F$ with pairwise distinct valuations under some discrete valuation v are linearly independent over the base field $\mathbb{F}_q$.

As its name suggests, one should think of the elements of a function field as functions. These functions are evaluated on *places*. Every discrete valuation induces a *place* $P \triangleq \{f \in F \mid v(f) > 0\}$ and also a *valuation ring* $O \triangleq \{f \in F \mid v(f) \ge 0\}$. P is a maximal ideal of O and so $F_P \triangleq O/P$ is a field. For $f \in O$, the coset of f in F_P is denoted by $f(P)$. This defines an embedding of $\mathbb{F}_q$ into F_P. The degree of the place P, denoted by $\deg(P)$, is defined as $[F_P : \mathbb{F}_q]$. A place is called *rational* if $\deg(P) = 1$. In such a case, F_P is isomorphic to $\mathbb{F}_q$.

The number of rational places of $F/\mathbb{F}_q$ is finite and is denoted by $N(F)$. If $v(f) < 0$ then the place P induced by v is called a *pole* of f. If $v(f) > 0$ then P is called a *zero* of f. A *divisor* is a formal sum of places $D = \sum_P n_P P$, where n_P is non-zero only for finitely many places P. We write $v_P(D) = n_P$. The degree of a divisor is defined by

$\deg(D) \triangleq \sum_P n_P \deg P$. For two divisors D, E, we write $D \geq E$, if $v_P(D) \geq v_P(E)$ for all places P. The principal divisor of a non-zero function $f \in F$, denoted by (f), is defined as $(f) \triangleq \sum_P v_P(f) P$. It can be shown that this is indeed a divisor. Moreover, the degree of a principal divisor is always 0. The *pole divisor* is defined by $(f)_\infty \triangleq \sum_{P: v_P(f)<0} v_P(f) P$.

With every divisor D one can associate a vector space called the Riemann-Roch space, denoted by $\mathcal{L}(D) \triangleq \{f \mid (f) \geq -D\} \cup \{0\}$. The following relation between the degree of a divisor D and the dimension of $\mathcal{L}(D)$ is of central importance in the theory of algebraic function fields:

$$1 + \deg(D) - g(F) \leq \mathsf{dim}(\mathcal{L}(D)) \leq 1 + \deg(D),$$

where $g(F)$ is independent of D and is an invariant of the function field $F/\mathbb{F}_q$, called the *genus*. Moreover, for any divisor D with $\deg(D) \geq 2g(F) - 1$ it holds that $\mathsf{dim}(\mathcal{L}(D)) = 1 + \deg(D) - g(F)$.

The Garcia-Stichtenoth tower

Let p be a prime power, and $q = p^2$. The Garcia-Stichtenoth tower $\mathcal{F} = (F_0, F_1, F_2, \ldots)$ is defined as follows. $F_0 = \mathbb{F}_q(x_0)$, and for all $k > 0$, $F_k = F_{k-1}(x_k)$, where $x_k^p + x_k = \frac{x_{k-1}^p}{1+x_{k-1}^{p-1}}$. This tower was introduced and analyzed in [GS96]. Further analysis appears in [SAK+01] and [GS07, Chapter 1, Sec 5.1]. The material presented here is based on these two latter sources.

Let P_∞ be the unique pole of x_0 in $F_0 = \mathbb{F}_q(x_0)$. The place P_∞ totally ramifies at all levels, and $P_\infty^{(k)}$ denotes the unique place above P_∞ in F_k. The valuation associated with the place $P_\infty^{(k)}$ is denoted by $v_\infty^{(k)}$. It holds that $\deg(P_\infty^{(k)}) = 1$, thus, $v_\infty^{(k)}(x_k) = -1$. Moreover, $v_\infty^{(k)}(x_i) = -p^{k-i}$. The exact number of rational places in the Garcia-Stichtenoth tower is known, and is given by the formula

$$N(F_k) = \begin{cases} (p-1)p^{k+1} + 2p & \text{for odd } p,\ k \geq 2, \\ (p-1)p^{k+1} + 2q & \text{for even } p,\ k \geq 2. \end{cases}$$

The genus of F_k, denoted by $g(F_k)$, is about p^{k+1}. An exact formula for the genus is given by

$$g(F_k) = \begin{cases} (p^{(k+1)/2} - 1)^2 & \text{for odd } k, \\ (p^{k/2} - 1)(p^{k/2+1} - 1) & \text{for even } k. \end{cases}$$

$\mathcal{F}$ is an asymptotically optimal tower as it achieves the Drinfeld-Vladut bound. In fact, for all k, $N(F_k)/g(F_k) \geq p - 1$.

3 Fooling Sums with Bounded Coefficients over Prime Modulo

In this section we introduce and construct a pseudorandom object that we use for the construction of our hitting set generators.

Definition 3.1. *A sequence of integers $(a_1, \ldots, a_n)$ is called d-bounded if each $a_i \in [-d, d]$ and there are at most d non-zero a_i's. Let $n, m, d \in \mathbb{N}$. A sequence of random variables $(Z_1, \ldots, Z_n)$ supported on $\{0, 1, \ldots, m-1\}^n$ is called* ε-biased for linear tests modulo m with d-bounded coefficients, *if for all non-zero d-bounded sequences $(a_1, \ldots, a_n)$, it holds that*

$$\mathbf{Pr}\left[\sum_{i=1}^{n} a_i Z_i = 0 \pmod{m}\right] \le \varepsilon.$$

An explicit construction of ε-biased distribution $(Z_1, \ldots, Z_n)$ for linear tests modulo m with *unbounded* coefficients (i.e., $d = m$) is implicit in the work of Alon and Mansour [AM95] and Azar, Motwani and Naor [AMN98]. In particular, the bias obtained is $\varepsilon = n/m$ and the seed length used is $O(\log m)$. Influenced by ideas from [AM95] and [NN93], we obtain the following.

Theorem 3.2. *There exists an algorithm that given n, d, and a prime number $m > 2d^3$ as inputs, outputs $(Z_1, \ldots, Z_n)$ which is $2d\log(n)/m$-biased for linear tests modulo m with d-bounded coefficients. The algorithm uses $\log m$ random bits and outputs $(Z_1, \ldots, Z_n)$ in time* $\mathrm{poly}(n, d, \log m)$.

In the proof of Theorem 3.2 we make use of BCH codes over non-binary alphabets. More precisely, we are interested in their parity check matrices. More information can be found in standard text books on coding theory, such as [Rot06] (see also Appendix B in [Bog05]).

Theorem 3.3. *Let p be a prime number and let $\ell, d \in \mathbb{N}$. Then, there exists an $\ell d \times p^\ell$ matrix A over $\mathbb{F}_p$ such that every d columns of A are linearly independent over $\mathbb{F}_p$. Moreover, A can be computed in time* $\mathrm{poly}(d \cdot p^\ell)$.

With this we are ready to prove Theorem 3.2.

of Theorem 3.2. Let $\varepsilon = 2d\log(n)/m$.

- Let D be the least prime number strictly larger than d. Note that $D \le 2d$. Let ℓ be the least integer such that $D^\ell \ge n$. Consider the $\ell D \times D^\ell$ matrix A over $\mathbb{F}_D$ from Theorem 3.3, and let B be the leftmost $\ell D \times n$ submatrix of A. Denote the i^{th} column of B by B_i, where we view it as a vector over the integers, $B_i \in \mathbb{Z}^{\ell D}$, with entries between 0 and $D - 1$.

- Let $f_1, \ldots, f_{\varepsilon m}$ be a basis for the $[m, \varepsilon m, (1-\varepsilon)m]_m$ Reed Solomon code (e.g., $f_i(x) = x^{i-1}$ for $i = 1, \ldots, \varepsilon m$). Define the sequence of random variables $(Y_1, \ldots, Y_{\varepsilon m})$, where each Y_i is supported on $\mathbb{F}_m$, as follows. Pick $r \sim [m]$ uniformly at random, and let $Y_i = f_i(r)$. One can verify that $\ell D \le \varepsilon m$ and define $Y = (Y_1, \ldots, Y_{\ell D}) \in \mathbb{F}_m^{\ell D}$.

We define the sequence of random variables $(Z_1, \ldots, Z_n)$ as follows. For $i \in [n]$, $Z_i = \langle B_i, Y \rangle$, where we consider the entries of B_i, Y (which are taken from $\mathbb{Z}$ and $\mathbb{F}_m$ respectively) as elements of $\mathbb{F}_m$, and perform addition and multiplication over $\mathbb{F}_m$. In particular, each Z_i is supported on $\{0, 1, \ldots, m-1\}$.

Let $(a_1, \dots, a_n)$ be a non-zero d-bounded sequence. Consider the random variable $X = a_1 Z_1 + \cdots + a_n Z_n$, where the sum is over $\mathbb{F}_m$. Our goal is to show that $\mathbf{Pr}[X = 0 \pmod{m}] \le \varepsilon$. By the definition of the Z_i's,

$$X = a_1 \langle B_1, Y \rangle + \cdots + a_n \langle B_n, Y \rangle = \langle a_1 B_1 + \cdots + a_n B_n, Y \rangle.$$

Since at most $d < D$ of the a_i's are non-zero integers, and since each a_i is in $[-d, d]$, it follows that at most D of the a_i's are non-zero over $\mathbb{F}_D$. Thus, by the definition of B, $a_1 B_1 + \cdots + a_n B_n$ is a non-zero vector in $\mathbb{F}_D^{\ell D}$. Hence, $C = (c_1, \dots, c_{\ell D}) = a_1 B_1 + \cdots + a_n B_n \in \mathbb{Z}^{\ell D}$ is a non-zero vector over the integers. That is, for some j, $c_j = \sum_i a_i (B_i)_j \neq 0$ as an integer. However, $|c_j| \le d \cdot d \cdot D \le 2d^3 < m$. This is because there are at most d summands with non-zero a_i, for each such summand $|a_i| \le d$ and $|(B_i)_j| \le D$. Hence $C \pmod{m}$ is a non-zero vector in $\mathbb{F}_m^{\ell D}$.

Thus, $X = \langle a_1 B_1 + \cdots + a_n B_n, Y \rangle = \sum c_i f_i(r)$ is an element of $\mathbb{F}_m$ obtained by sampling uniformly a random entry r of the non-zero codeword $\sum c_i f_i$ of a Reed Solomon code with relative distance $1 - \varepsilon$. In particular, $\mathbf{Pr}[X = 0 \pmod{m}] \le \varepsilon$.

The randomness used by the algorithm is for sampling $r \sim [m]$ uniformly at random, and so $\log m$ random bits are used. Computing D, the least prime number larger than d, can be done deterministically in time $\text{poly}(d)$. For computing the sequence $(Z_1, \dots, Z_n)$ one needs to compute the matrix B, which by Theorem 3.3, can be done in time $\text{poly}(D^\ell) = \text{poly}(n, d)$. Also Y is a column of the Reed Solomon generating matrix. After fixing r, each entry of Y can be computed by performing field operations over $\mathbb{F}_m$, where m is a prime. In particular, powering field elements of $\mathbb{F}_m^\times$ up to power bounded above by $\text{poly}(m)$. This can be done in time $\text{polylog}(m)$. Since there are $O(d \log n)$ entries in Y, the proof follows. $\square$

4 Hitting Set Generators for Relatively Small Fields

In this section we present our construction of a hitting set generator and states its proof of correctness in Theorem 4.1. Together with Bogdanov's reduction (Theorem 2.3) this readily implies Theorem 1.1.

Theorem 4.1. *For any $s, d \in \mathbb{N}$, $\delta > 0$ and $n \in \mathbb{N}$, let p be the least prime power larger than $\frac{4d}{\delta} + 1$* [6]*, and set $q = p^2$. Then, the algorithm in Figure 1 is a hitting set generator with density $1 - \delta$ for polynomials in $\mathbb{F}_q[y_1, \dots, y_n]$ with degree d and sparsity s. The number of random bits used by the algorithm is $4 \log(sd^2/\delta) + 2 \log \log n + O(1)$ and the running-time is* $\text{poly}(s, n, d, 1/\delta)$.

Proof. First note that the output $(f_1(P), \dots, f_n(P))$ is well-defined as $P \neq P_\infty^{(k)}$ and all f_i's lie in the Riemann-Roch space of some multiple of $P_\infty^{(k)}$. Moreover, $(f_1(P), \dots, f_n(P))$

[6]The theorem holds for any prime power p that is larger than $\frac{4d}{\delta} + 1$. However, in order not to introduce a dependency of p in the seed length (for simplicity sake), we consider the smallest p for which the theorem holds.

Hitting Set Generator for Relatively Small Fields

Input. $n, s, d \in \mathbb{N}$, $\delta > 0$ and the least prime power p such that $p \geq 4d/\delta + 1$. We denote $q = p^2$.

Output. A sample $(Y_1, \ldots, Y_n) \in \mathbb{F}_q^n$ such that for every non-zero polynomial $h \in \mathbb{F}_q[y_1, \ldots, y_n]$ with degree d and sparsity s, $\mathbf{Pr}[h(Y_1, \ldots, Y_n) = 0] \leq \delta$.

1. Find the smallest prime number $m > 16s^2d^3 \log(n)/\delta$, and let k be the lowest level in the Garcia-Stichtenoth tower such that $N(F_k) \geq 6md/\delta$.
2. Sample $(Z_1, \ldots, Z_n)$ from a $(4d \log(n)/m)$-biased distribution for linear tests modulo m with $2d$-bounded coefficients.
3. Let r be the smallest integer divisible by m, such that $r \geq 2g(F_k) - 1$.
4. For each $i \in [n]$, find $f_i \in \mathcal{L}((r + Z_i) \cdot P_\infty^{(k)}) \setminus \mathcal{L}((r + Z_i - 1) \cdot P_\infty^{(k)})$.
5. Sample uniformly at random a rational place $P \neq P_\infty^{(k)}$ of F_k.
6. Return $(f_1(P), \ldots, f_n(P))$.

Figure 1: A hitting set generator for relatively small fields.

is indeed in $\mathbb{F}_q^n$ as P is a rational place. Note also that for each $i \in [n]$, there exists $f_i \in \mathcal{L}((r + Z_i) \cdot P_\infty^{(k)}) \setminus \mathcal{L}((r + Z_i - 1) \cdot P_\infty^{(k)})$. This is because $r \geq 2g(F_k) - 1$ (and $Z_i \geq 0$).

Consider a monomial $y_1^{d_1} \cdots y_n^{d_n}$ of degree at most d, that is, $\sum_{i=1}^n d_i \leq d$. The algorithm maps any variable y_i to some function $f_i \in \mathcal{L}((r + Z_i) \cdot P_\infty^{(k)}) \setminus \mathcal{L}((r + Z_i - 1) \cdot P_\infty^{(k)})$. This induces a mapping of the monomial $y_1^{d_1} \cdots y_n^{d_n}$ to the function $f_1^{d_1} \cdots f_n^{d_n}$. We note that the latter function is in $\mathcal{L}(d(r + m) \cdot P_\infty^{(k)})$ as it has no poles outside $P_\infty^{(k)}$, and

$$v_\infty(f_1^{d_1} \cdots f_n^{d_n}) = \sum_{i=1}^n d_i v_\infty(f_i) = -\sum_{i=1}^n d_i(r + Z_i) \geq -\sum_{i=1}^n d_i(r + m) \geq -d(r + m).$$

Consider now two monomials $y_1^{d_1} \cdots y_n^{d_n}$, $y_1^{d'_1} \cdots y_n^{d'_n}$ of degree at most d. The claim is that, except for probability $4d \log(n)/m$, the two monomials are mapped to functions in F_k that have different valuations at $P_\infty^{(k)}$:

$$\begin{aligned}\mathbf{Pr}\left[v_\infty\left(f_1^{d_1}\cdots f_n^{d_n}\right)=v_\infty\left(f_1^{d_1'}\cdots f_n^{d_n'}\right)\right]&=\mathbf{Pr}\left[\sum_{i=1}^n d_i(r+Z_i)=\sum_{i=1}^n d_i'(r+Z_i)\right]\\&\le\mathbf{Pr}\left[\sum_{i=1}^n d_i(r+Z_i)=\sum_{i=1}^n d_i'(r+Z_i)\pmod m\right]\\&=\mathbf{Pr}\left[\sum_{i=1}^n (d_i-d_i')Z_i=0\pmod m\right],\end{aligned}$$

which is at most $4d\log(n)/m$. The third inequality follows since m divides r, and the last inequality follows since $(Z_1,\dots,Z_n)$ is sampled from a $(4d\log(n)/m)$-biased distribution for linear tests modulo m with $2d$-bounded coefficients, and indeed, there are at most $2d$ indices i such that d_i-d_i' is not zero, and for such i, $d_i-d_i'\in[-d,d]\subseteq[-2d,2d]$. Moreover, note that $m>2\cdot(2d)^3$, and thus the hypothesis of Theorem 3.2 does indeed hold.

Consider a polynomial $h\in\mathbb{F}_q[y_1,\dots,y_n]$ with degree d and sparsity s. By the union bound, the probability that there exist two monomials of h that are mapped to functions with the same valuation at $P_\infty^{(k)}$ is at most $\binom{s}{2}\cdot4d\log(n)/m$, which is at most $\delta/8$ by the choice of m. Hence, except for probability $\delta/8$, the polynomial h is mapped to a non-zero function $f\in\mathcal{L}(d(r+m)\cdot P_\infty^{(k)})$, as functions with different valuations are linearly independent.

We now restrict ourselves to the event where $f\neq0$ and show that $\mathbf{Pr}_P[f(P)=0]\le(7/8)\cdot\delta$, where $P\neq P_\infty^{(k)}$ is a rational place of F_k sampled uniformly at random. This will conclude the proof for the algorithm's correctness. Assume there are z rational places $P_1,\dots,P_z$ such that $f(P_i)=0$ for all $i\in[z]$. Let D be the divisor $d(r+m)P_\infty^{(k)}-(P_1+\cdots+P_z)$. Since $f(P_i)=0$ for all $i\in[z]$, $f\in\mathcal{L}(D)$. Since $f\neq0$, it follows that $\mathsf{dim}(\mathcal{L}(D))>0$ and so $\deg(D)\ge0$. However, $\deg(D)=d(r+m)-z$, which implies $z\le d(r+m)$. Thus,

$$\Pr_P[f(P)=0\mid f\neq0]=\frac{z}{N(F_k)}\le\frac{d(r+m)}{N(F_k)}\le\frac{2d(g(F_k)+m)}{N(F_k)}\le\frac{2d}{p-1}+\frac{3}{8}\cdot\delta\le\frac{7}{8}\cdot\delta,$$

where we used the fact that $r\le2g(F_k)+m$, the choice of m,p and the fact that $N(F_k)/g(F_k)\ge p-1$ for all k.

We now upper bound the number of random bits used. The algorithm performs two random steps. First there is the sampling of $(Z_1,\dots,Z_n)$, which by Theorem 3.2, requires $\log m$ random bits. Secondly, we sample a rational place uniformly out of $N(F_k)$ places. We chose k to be the least integer such that $N(F_k)\ge6md/\delta$ and so $N(F_{k-1})<6md/\delta$. Since $N(F_k)\le p\cdot N(F_{k-1})$, the number of random bits used is at most $\log(6mdp/\delta)$. The assertion regarding the number of random bits used readily follows.

We turn to analyze the running-time of the algorithm. Computing the prime number m can be carried out (deterministically) in time $\mathrm{poly}(m)$. By Theorem 3.2, sampling $(Z_1,\dots,Z_n)$ can be done in time $\mathrm{poly}(n,d,\log m)$. In order to find $f_i\in\mathcal{L}((r+Z_i)\cdot P_\infty^{(k)})\setminus\mathcal{L}((r+Z_i-1)\cdot P_\infty^{(k)})$, one can find a basis for $\mathcal{L}((r+Z_i)\cdot P_\infty^{(k)})$, in time $\mathrm{poly}(r)$ [SAK+01], and find in this basis an element that has valuation $r+Z_i$ at $P_\infty^{(k)}$, also in time $\mathrm{poly}(r)$. Since

$r \leq 2g(F_k) + m = O(md/\delta)$, this can be carried out in time $\mathrm{poly}(m)$. The last two steps of the algorithm can clearly be carried out in time $\mathrm{poly}(n, \log m)$. The total running-time is therefore $\mathrm{poly}(s, n, d, 1/\delta)$.

By examining the arguments above, one can see that the bottleneck in the running time is determined by finding the prime number $m \sim s^2/\delta$, and finding a function f with a given valuation. These are the only steps that cost us the $\mathrm{poly}(s/\delta)$ in the running time. It turns out that finding m can be done in time $\mathrm{polylog}(m)$ probabilistically (see Section 5.2), and we chose to present the simpler naive algorithm for finding a prime above, which runs in time $\mathrm{poly}(m)$, for simplicity. Finding a function f with a given valuation is the real bottleneck in the running time. In Section 5 we bypass this problem, however, our solution costs us in the field size. Clearly, if Conjecture 1.4 holds, then finding f can be done in time $\mathrm{polylog}(s/\delta)$, and the algorithm described above runs in time $\mathrm{poly}(n, d, \log(1/\delta))$. $\square$

5 The More Efficient Hitting Set Generator

The More Efficient Hitting Set Generator

Input. $n, s, d \in \mathbb{N}$, a prime power p and $\delta > 0$ such that $p = \Omega\left(\frac{d \cdot \log(ds \log(n)/\delta)}{\delta \cdot \log\log s}\right)$. We denote $q = p^2$.

Output. A sample $(Y_1, \ldots, Y_n) \in \mathbb{F}_q^n$ such that for every non-zero polynomial $h \in \mathbb{F}_q[y_1, \ldots, y_n]$ of degree d and sparsity s, $\mathbf{Pr}[h(Y_1, \ldots, Y_n) = 0] \leq \delta$.

1. Find some prime number $m \in [16d^3s^2 \log(n)/\delta, 32d^3s^2 \log(n)/\delta]$ with failure probability bounded by $\delta/2$. Let k be the least integer such that $p^k \geq 2m$ and F_k the kth function field in Garcia-Stichtenoth tower.
2. Sample $(Z_1, \ldots, Z_n)$ from a $(4d \log(n)/m)$-biased distribution for linear tests modulo m with $2d$-bounded coefficients.
3. Let r be the least integer divisible by m such that $r > k \cdot p^{k+1} - \frac{p^k - 1}{p-1} - p^k$.
4. For each $i \in [n]$, find $f_i \in \mathcal{L}((r + Z_i) \cdot P_\infty^{(k)}) \setminus \mathcal{L}((r + Z_i - 1) \cdot P_\infty^{(k)})$.
5. Sample uniformly at random a rational place $P \neq P_\infty^{(k)}$ of F_k.
6. Return $(f_1(P), \ldots, f_n(P))$.

Figure 2: The more efficient hitting set generator.

In this section we prove the following theorem.

Theorem 5.1. *Let p be a prime power, and let $q = p^2$. For any $s, d \in \mathbb{N}$, $\delta > 0$ and $n \in \mathbb{N}$, such that*

$$p = \Omega\left(\frac{d \cdot \log\left(ds \log\left(n\right)/\delta\right)}{\delta \cdot \log\log s}\right),$$

the following holds. The algorithm in Figure 2 is a hitting set generator with density $1-\delta$ for polynomials in $\mathbb{F}_q[y_1, \ldots, y_n]$ with degree d and sparsity s. The number of random bits used by the algorithm is $8\log(sd^2/\delta) + 3\log\log n + O(1)$ and the running-time is $\mathrm{poly}(n, d, \log(s/\delta))$.

Theorem 1.2 readily follows by Theorem 5.1 and Bogdanov's reduction (Theorem 2.3). To prove Theorem 5.1, we prove two things:

- One can efficiently find elements $f_i \in \mathcal{L}((r+Z_i)\cdot P_\infty^{(k)}) \setminus \mathcal{L}((r+Z_i-1)\cdot P_\infty^{(k)})$ for many i's. This follows from Theorem 1.5 whose proof is given in Section 5.1.
- One can find appropriate primes efficiently using a *randomized* algorithm. This is stated in Lemma 5.3. This fact is known [pol09], and we give it an alternative proof in Section 5.2, with slightly improved randomness complexity.

We then complete the proof of Theorem 5.1 in Section 5.3.

5.1 Proof of Theorem 1.5

Let F_k be the kth level in the Garcia-Stichtenoth tower over $\mathbb{F}_q$. For $0 \le i \le k$, define $\pi_i = \prod_{j=0}^{i}(x_j^{p-1}+1)$, and let

$$u_{i,e} = \begin{cases} x_i^e, & \text{if } \ 0 \le e < p-1; \\ x_i^e + 1, & \text{if } \ e = p-1. \end{cases}$$

In [PST98] (see their Lemma 3.4, ii), it is shown that each function in

$$\{\pi_{i-1} \cdot u_{i,e} \,|\, 0 \le i \le k\,, \quad 0 \le e \le p-1\}$$

has poles only at $P_\infty^{(k)}$ in F_k. Moreover (see Table I in [SAK+01]), $v_\infty^{(k)}(\pi_i) = -(p^{k+1} - p^{k-i})$ and $v_\infty^{(k)}(x_i) = -p^{k-i}$. Thus, $v_\infty^{(k)}(u_{i,e}) = -e \cdot p^{k-i}$. Given that, we conclude the following lemma, which readily implies Theorem 1.5.

Lemma 5.2. *For every integer $k \ge 1$ the following holds. Let $0 \le a \le p^k - 1$ be some integer that is written as $a_0 + a_1 \cdot p + a_2 \cdot p^2 + \cdots + a_{k-1} \cdot p^{k-1}$ in base p (namely, $0 \le a_i \le p-1$ for $i = 0, 1, \ldots, k-1$). Define the function*

$$f_a = \prod_{i=1}^{k} \pi_{k-i} \cdot u_{k-i+1, p-1-a_{i-1}}.$$

Then, in F_k, f_a has poles only at $P_\infty^{(k)}$, and

$$-v_\infty^{(k)}(f_a) = k \cdot p^{k+1} - \frac{p^k - 1}{p-1} - a. \tag{2}$$

Moreover, given p, k and $0 \leq a \leq p^k - 1$, the function f_a can be computed and evaluated on a given place in time $\mathrm{polylog}(p^k)$.

Proof. First note that by the above discussion, the only poles f_a has in F_k is at $P_\infty^{(k)}$. For fixed $1 \leq i \leq k$,

$$\begin{aligned} -v_\infty^{(k)}(\pi_{k-i} \cdot u_{k-i+1,p-1-a_{i-1}}) &= p^{k+1} - p^i + (p-1-a_{i-1}) \cdot p^{i-1} \\ &= p^{k+1} - p^{i-1} - a_{i-1} \cdot p^{i-1}. \end{aligned}$$

Equation (2) then follows by taking the sum, over $i = 1, \ldots, k$, of the right hand side in the equation above. The assertion regarding the running-time for computing f_a is trivial. □

5.2 Finding a Prime in an Interval using Random Bits

The algorithm in Figure 2 involves finding a prime number m larger than $\Omega(s^2)$, for the construction of the small-bias sequence that fools linear tests modulo m. Since we are willing to run only for $\mathrm{polylog}(s)$ time, we cannot settle for known deterministic algorithms. However, we do have few random bits at our disposal, and it is known how to find m as above with success probability $1 - \delta$ using $O(\log(s/\delta))$ random bits [pol09]. We give an alternative proof for this known fact in Lemma 5.3 below. Also, we slightly improve upon the randomness complexity of known techniques [7] (one uses the Nisan-Zuckerman pseudorandom generator [NZ96], and the other is based on an analysis of the moments of the indicator random variable for being a prime number in a given interval, via sieve theory).

Lemma 5.3 (Finding a prime in an interval)**.** *There exists a randomized algorithm that given m and $\delta > 0$ as inputs, outputs a prime number $p \in [m, 2m]$ with probability at least $1 - \delta$. The running-time of the algorithm is* $\mathrm{polylog}(m) \cdot \log(1/\delta)$ *and the number of random bits used by the algorithm is* $0.525 \cdot \log m + (2 + o(1)) \cdot \log(1/\delta) + O(1)$.

Proof. It is well-known that in the range $[m, 2m]$ there are $\Omega(m/\log m)$ prime numbers. However, better results are known. Define $\Gamma(m) = m^{0.525}$. Baker *et al.* [BHP01] showed that for large enough m, in the interval $[m, m + \Gamma(m)]$, at least $9/(100 \cdot \log m)$ fraction of the numbers are primes. [8]

Testing whether a given integer x is a prime can be done deterministically in time $\mathrm{polylog}(x)$ [AKS04]. Thus, a first attempt to find a prime number in the interval $[m, m + \Gamma(m)]$ would simply be to sample uniformly $x \sim [m, m + \Gamma(m)]$ and test whether x is prime. Finding a prime in one iteration succeed with probability $\Omega(1/\log m)$. Thus, repeating this process for $O(\log(m) \cdot \log(1/\delta))$ iterations reduces the failure probability to δ. Since we use $\log \Gamma(m)$ random bits per iteration, the total number of random bits used by this algorithm

[7] We have not been able to find a formal proof for known techniques in the literature, and specifically we do not know of an exact assertion regarding the number of random bits used, but doing some calculation, it seems our method does better in terms of randomness complexity.

[8] In fact, the main theorem of Baker *et al.* [BHP01] only states the existence of a prime in $[m, m + \Gamma(m)]$. Nevertheless, they actually proved the stronger claim we need.

is $O(\log^2(m)\cdot\log(1/\delta))$. Furthermore, the running-time is $\mathrm{polylog}(m)\cdot\log(1/\delta)$ as we can test the primality of each sample, deterministically, in time $\mathrm{polylog}(m)$.

One can reduce the number of random bits by using a hitter. Slightly deviating from the standard notation for our own needs, a hitter for density μ and error parameter δ is a randomized algorithm that gets as inputs m, k, δ, μ, and outputs a (random) set $H \subseteq [m, m+k]$ with the following property. For every set $B \subseteq [m, m+k]$ with density $|B|/(k+1) \geq \mu$, it holds that $\mathbf{Pr}_H[H \cap B = \emptyset] \leq \delta$. The maximum size of H that the hitter outputs is called the *sample complexity* of the hitter.

In [Gol97], Corollary C.5, a hitter with sample complexity $O(\log(1/\delta)/\mu)$ is presented. The number of random bits used by this hitter is $\log k + (2+o(1))\cdot\log(1/\delta)$. By plugging $k = \Gamma(m)$ and $\mu = \Omega(1/\log m)$ we get the desired parameters. $\square$

5.3 Proof of Theorem 5.1

Proof. As in the proof of Theorem 4.1, we note that the output $(f_1(P), \ldots, f_n(P))$ is well-defined and is contained in $\mathbb{F}_q^n$. Moreover, since $p^k \geq 2m$ and r is the least integer divisible by m such that $r > k\cdot p^{k+1} - \frac{p^k-1}{p-1} - p^k$, there are at least m integers in the range $[r, kp^{k+1} - \frac{p^k-1}{p-1}]$. This is necessary as the algorithm needs to find a function with a given valuation v at $P_\infty^{(k)}$, for v's in $\{r, r+1, \ldots, r+m-1\}$. Doing so efficiently we only know how to do inside the interval $[kp^{k+1} - \frac{p^k-1}{p-1} - p^k + 1, kp^{k+1} - \frac{p^k-1}{p-1}]$.

Consider a monomial $y_1^{d_1}\cdots y_n^{d_n}$ of degree at most d. The algorithm maps every variable y_i to some function $f_i \in \mathcal{L}((r+Z_i)\cdot P_\infty^{(k)}) \setminus \mathcal{L}((r+Z_i-1)\cdot P_\infty^{(k)})$. This induces a mapping of the monomial $y_1^{d_1}\cdots y_n^{d_n}$ to the function $f_1^{d_1}\cdots f_n^{d_n}$. We note that the latter function is in $\mathcal{L}(dkp^{k+1}\cdot P_\infty^{(k)})$ as it has no poles outside $P_\infty^{(k)}$, and

$$v_\infty(f_1^{d_1}\cdots f_n^{d_n}) = \sum_{i=1}^n d_i v_\infty(f_i) \geq -\sum_{i=1}^n d_i kp^{k+1} \geq -dkp^{k+1}.$$

Consider now two monomials $y_1^{d_1}\cdots y_n^{d_n}$, $y_1^{d'_1}\cdots y_n^{d'_n}$ of degree at most d. The claim is that, except with probability $4d\log(n)/m$, the two monomials are mapped to functions in F_k that have different valuations at $P_\infty^{(k)}$. Thus, as in the proof of Theorem 4.1, the probability that two distinct monomials of a polynomial $h \in \mathbb{F}_q[y_1, \ldots, y_n]$ with degree d and sparsity s will be mapped to functions with the same valuation is at most $\binom{s}{2}\cdot\frac{4d\log n}{m}$, which is at most $\delta/8$ by the choice of m. We thus conclude that except with probability $\delta/8$, the polynomial h is mapped to a non-zero function $f \in \mathcal{L}(dkp^{k+1}\cdot P_\infty^{(k)})$.

We now restrict ourselves to the event where $f \neq 0$. As before it is enough to show that $\mathbf{Pr}_P[f(P) = 0] \leq \delta/4$, where $P \neq P_\infty^{(k)}$ is a rational place of F_k sampled uniformly at random. This will conclude the proof for the algorithm's correctness. By the same argument used in the proof of Theorem 4.1, since $0 \neq f \in \mathcal{L}(dkp^{k+1}\cdot P_\infty^{(k)})$, it follows that

$$\mathop{\mathbf{Pr}}_P[f(P) = 0 \mid f \neq 0] = \frac{dkp^{k+1}}{N(F_k)} < \frac{dk}{p-1} \leq \frac{\delta}{4},$$

where the second inequality follows as $N(F_k) > (p-1)p^{k+1}$. The last inequality follows, for large enough n, by our choice of p and since k is chosen such that $p^{k-1} < 2m$.

We now upper bound the number of random bits used. The algorithm performs three random steps. First, by Lemma 5.3, finding the prime number m, with failure probability at most $\delta/2$, can be done using $0.525 \cdot \log m + (2+o(1)) \cdot \log(2/\delta) + O(1)$ random bits. Secondly, by Theorem 3.2, sampling $(Z_1, \ldots, Z_n)$ requires $\log m$ random bits. Third, the number of rational places $N(F_k)$ is bounded above by $p^{k+2} < 2p^3 m$ and so sampling a rational place costs $\log m + \log(2p^3)$ random bits. It can be easily verified that the assertion regarding the number of random bits used holds.

We now analyze the running-time of the algorithm. By Lemma 5.3, computing m can be done in time $\mathrm{polylog}(m/\delta)$. By Theorem 3.2, sampling $(Z_1, \ldots, Z_n)$ can be carried out in $\mathrm{poly}(n, d, \log m)$ time. By Lemma 5.2, finding a function f_i with a given valuation can be done in time $\mathrm{polylog}(m)$. Finally, sampling a place and evaluating the computed function on it, can be done in time $\mathrm{polylog}(m)$ as well. Thus, the total running-time is $\mathrm{poly}(n, d, \log(m/\delta)) = \mathrm{poly}(n, d, \log(s/\delta))$ $\square$

Acknowledgement

We thank Venkatesan Guruswami and Chaoping Xing for bringing to our attention the fact that Lu's hitting set generator yields a pseudorandom generator via Bogdanov's reduction. The first author would like to thank his advisor Ran Raz for his continuous encouragement.

References

[ABN+92] N. Alon, J. Bruck, J. Naor, M. Naor, and R. Roth. Construction of asymptotically good low-rate error-correcting codes through pseudo-random graphs. *IEEE Transactions on Information Theory*, 38:509–516, 1992.

[AGHP92] N. Alon, O. Goldreich, J. Håstad, and R. Peralta. Simple construction of almost k-wise independent random variables. *Random Structures and Algorithms*, 3(3):289–304, 1992.

[AKS04] M. Agrawal, N. Kayal, and N. Saxena. Primes is in P. *Annals of Mathematics*, 160(2):781–793, 2004.

[AM95] N. Alon and Y. Mansour. ε-discrepancy sets and their application for interpolation of sparse polynomials. *Information Processing Letters*, 54(6):337–342, 1995.

[AMN98] Y. Azar, R. Motwani, and J. Naor. Approximating probability distributions using small sample spaces. *Combinatorica*, 18(2):151–171, 1998.

[BHP01] R. C. Baker, G. Harman, and J. Pintz. The difference between consecutive primes, II. *Proceedings of the London Mathematical Society*, 83:532–562, 2001.

[Bog05] A. Bogdanov. Pseudorandom generators for low degree polynomials. In *Proceedings of the 37th annual STOC*, pages 21–30, 2005.

[BT09] A. Ben-Aroya and A. Ta-Shma. Constructing small-bias sets from algebraic-geometric codes. In *Proceedings of the 50th annual IEEE symposium on foundations of computer science (FOCS)*, pages 191–197, 2009.

[BV10] A. Bogdanov and E. Viola. Pseudorandom bits for polynomials. *SIAM J. Comput.*, 39(6):2464–2486, 2010.

[Gol97] O. Goldreich. A sample of samplers - a computational perspective on sampling (survey). *Electronic Colloquium on Computational Complexity (ECCC)*, 4(20), 1997.

[Gop81] V. D. Goppa. Codes on algebraic curves. In *Soviet Math. Dokl*, volume 24, pages 170–172, 1981.

[GS96] A. Garcia and H. Stichtenoth. On the asymptotic behaviour of some towers of function fields over finite fields. *Journal of Number Theory*, 61(2):248–273, 1996.

[GS07] A. Garcia and H. Stichtenoth. *Topics in geometry, coding theory and cryptography*, volume 6. Springer, 2007.

[GT07] B. Green and T. Tao. The distribution of polynomials over finite fields, with applications to the Gowers norms. arXiv:0711.3191, 2007.

[GT08] B. Green and T. Tao. An inverse theorem for the Gowers U^3-norm, with applications. *Proc. Edinburgh Math. Soc.*, 51(1):73–153, 2008.

[GX13] V. Guruswami and C. Xing. Hitting sets for low-degree polynomials with optimal density. *ECCC*, (175), 2013.

[Heß02] F. Heß. Computing Riemann–Roch spaces in algebraic function fields and related topics. *Journal of Symbolic Computation*, 33(4):425–445, 2002.

[KL08] T. Kaufman and S. Lovett. Worst case to average case reductions for polynomials. In *Foundations of Computer Science (FOCS), 2008 49th Annual IEEE Symposium on*, pages 166–175. IEEE, 2008.

[KS01] A. Klivans and D. Spielman. Randomness efficient identity testing of multivariate polynomials. In *Proceedings of the 33rd Annual STOC*, pages 216–223, 2001.

[LMS08] S. Lovett, R. Meshulam, and A. Samorodnitsky. Inverse conjecture for the Gowers norm is false. In *40th Annual STOC*, pages 547–556, 2008.

[Lov08] S. Lovett. Unconditional pseudorandom generators for low degree polynomials. In *40th Annual STOC*, pages 557–562, 2008.

[Lu12] C. J. Lu. Hitting set generators for sparse polynomials over any finite fields. In *Computational Complexity (CCC), 2012 IEEE 27th Annual Conference on*, pages 280–286. IEEE, 2012.

[LVW93] M. Luby, B. Veličković, and A. Wigderson. Deterministic approximate counting of depth-2 circuits. In *Theory and Computing Systems, 1993., Proceedings of the 2nd Israel Symposium on the*, pages 18–24. IEEE, 1993.

[NN93] J. Naor and M. Naor. Small-bias probability spaces: Efficient constructions and applications. *SIAM J. on Computing*, 22(4):838–856, 1993.

[NZ96] N. Nisan and D. Zuckerman. Randomness is linear in space. *J. of Computer and System Sciences*, 52(1):43–52, 1996.

[pol09] Finding primes (polymath 4), 2009. http://michaelnielsen.org/polymath1/index.php?title=Finding_primes_with_O(k)_random_bits.

[PST98] R. Pellikaan, H. Stichtenoth, and F. Torres. Weierstrass semigroups in an asymptotically good tower of function fields. *Finite fields and their applications*, 4(4):381–392, 1998.

[Rot06] R. M. Roth. Introduction to coding theory. *IET Communications*, page 47, 2006.

[SAK$^+$01] K. W. Shum, I. Aleshnikov, P. V. Kumar, H. Stichtenoth, and V. Deolalikar. A low-complexity algorithm for the construction of algebraic-geometric codes better than the Gilbert-Varshamov bound. *Information Theory, IEEE Transactions on*, 47(6):2225–2241, 2001.

[Sti93] H. Stichtenoth. *Algebraic Function Fields and Codes*. Universitext, Springer-Verlag, Berlin, 1993.

[Vio07] E. Viola. Pseudorandom bits for constant-depth circuits with few arbitrary symmetric gates. *SIAM Journal on Computing*, 36(5):1387–1403, 2007.

[Vio09a] E. Viola. Guest column: correlation bounds for polynomials over {0,1}. *ACM SIGACT News*, 40(1):27–44, 2009.

[Vio09b] E. Viola. The sum of d small-bias generators fools polynomials of degree d. *Computational Complexity*, 18(2):209–217, 2009.

	seed length	minimum field size	remarks
optimal	$d \log n$	2	
[LVW93]	$2^{O(\sqrt{n})}$	2	any constant degree
[Bog05]	$d^4 \log n$	$d^{10} \log^2 n$	
[Bog05]	$c^2 d^8 n^{6/(c-2)}$	cd^6	Each output element depends on c inputs.
[BV10]	$\log n$	2	$d = 2, 3$
[BV10]+[GT08]	$d \log n + f(d)$	$d + 1$	$f(d) \gg \exp(d)$
[BV10]+[KL08]	$d \log n + f(d)$	2	$f(d) \gg \exp(d)$
[Lov08]	$2^{O(d)} \log n$	2	
[Vio09b]	$d \log n + d2^d$	2	
[Lu12]	$(d^4/c) \cdot \log n$	d^{6+c}	For any $c > 0$
[Lu12] (special case)	$d^4 \cdot \log(d) \cdot \log(n)$	d^6	
[GX13]	$d^4 \cdot \log n$	d^6	
Theorem 1.1	$d^4 \cdot \log n$	d^{12}	Running time $n^{O(d^4)}$, but $n^{O(1)}$ assuming Conjecture 1.4.
Theorem 1.2	$d^4 \log n$	$d^{20} \log^{2-o(1)} n$	

Table 1: Explicit constructions of pseudorandom generators for low degree polynomials. For simplicity, we consider constant bias.