Electronic Colloquium on Computational Complexity, Revision 1 of Report No. 65 (2019)

Derandomization from Algebraic Hardness: Treading the Borders

Mrinal Kumar* Ramprasad Saptharishi† Noam Solomon‡

Abstract

A hitting-set generator (HSG) is a polynomial map $\mathsf{Gen} : \mathbb{F}^k \to \mathbb{F}^n$ such that for all n-variate polynomials Q of small enough circuit size and degree, if Q is non-zero, then $Q \circ \mathsf{Gen}$ is non-zero. In this paper, we give a new construction of such a HSG assuming that we have an explicit polynomial of sufficient hardness in the sense of *approximative or border complexity*. Formally, we prove the following result over any characteristic zero field $\mathbb{F}$:

> Suppose $P(z_1, \ldots, z_k)$ is an explicit k-variate degree d polynomial that is not in the *border of circuits* of size s. Then, there is an explicit hitting-set generator $\mathsf{Gen}_P : \mathbb{F}^{2k} \to \mathbb{F}^n$ such that every non-zero n-variate degree D polynomial $Q(\mathbf{x})$ in the border of size s' circuits satisfies $Q \neq 0 \Rightarrow Q \circ \mathsf{Gen}_P \neq 0$, provided $n^{10k} dDs' < s$.

This is the first HSG in the algebraic setting that yields a complete derandomization of polynomial identity testing (PIT) for general circuits from a suitable algebraic hardness assumption.

As a direct consequence, we show that even a slightly non-trivial explicit construction of hitting sets for polynomials in the border of constant-variate circuits implies a deterministic polynomial time algorithm for PIT.

> Let $\delta > 0$ be any constant and k be a large enough constant. Suppose, for every $s \geq k$, there is an explicit hitting set of size $s^{k-\delta}$ for all degree s polynomials in the border of k-variate size s algebraic circuits. Then, there is an explicit hitting set of size $\mathrm{poly}(s)$ for the border s-variate algebraic circuits of size s and degree s.

Unlike the prior constructions of such maps [NW94, KI04, AGS18, KST19], our construction is purely algebraic and does *not* rely on the notion of combinatorial designs.

*mrinalkumar08@gmail.com. Department of Computer Science, University of Toronto, Canada. Part of this work was done while at the Simons Institute for the Theory of Computing, Berkeley during the semester on Lower Bounds in Computational Complexity in Fall 2018.

†ramprasad@tifr.res.in. Tata Institute of Fundamental Research, Mumbai, India. Research supported by Ramanujan Fellowship of DST.

‡noam.solom@gmail.com. Department of Mathematics, MIT, Cambridge, MA, USA.

ISSN 1433-8092

1 Introduction

The interaction of hardness and randomness is one of the most well studied themes in computational complexity theory, and in this work we focus on exploring this interaction further in the realm of algebraic computation. To set the stage, we start with a brief introduction to algebraic complexity.

The field of algebraic complexity primarily focuses on studying multivariate polynomials and their complexity in terms of the number of basic operations (additions and multiplications) required to compute them. Algebraic circuits (which are just directed acyclic graphs with leaves labelled by variables or field constant, and internal gates labelled by $+$ or $\times$) form a very natural model of computation in this setting, and the size (number of gates or wires) of the smallest algebraic circuit computing a polynomial gives a robust measure of its complexity.

The main protagonists in the hardness-randomness interaction in algebraic complexity are the *hardness* component which is the question of proving superpolynomial lower bounds for algebraic circuits for any explicit polynomial family and the *randomness* component which is the question of designing efficient *deterministic* algorithms for polynomial identity testing (PIT) — the algorithmic task of checking if a given circuit computes the zero polynomial. Both these questions are of fundamental importance in computational complexity and are algebraic analogues of their more well known Boolean counterparts, the P vs NP question and the P vs BPP question respectively. These seemingly different problems are closely related to each other, and in this work we focus on one direction of this relationship; namely, the use of hard explicit polynomial families for derandomization of PIT.

It is known from an influential work of Kabanets and Impagliazzo [KI04] that lower bounds on the algebraic circuit complexity of explicit polynomial families leads to non-trivial deterministic algorithm for the question of polynomial identity testing (PIT) of algebraic circuits. Moreover, the results in [KI04] show that stronger lower bounds give faster deterministic algorithms for polynomial identity testing. For instance, from truly exponential (or $2^{\Omega(n)}$) lower bounds , we get quasipolynomial (or $n^{O(\log n)}$) time deterministic algorithms for PIT. From weaker superpolynomial (or $n^{\omega(1)}$) lower bounds, we only seem to get a subexponential (or $2^{n^{o(1)}}$) time PIT algorithm.

However, no matter how good the lower bounds for algebraic circuits are, this connection between lower bounds and derandomization does not seem to give truly polynomial time deterministic algorithms for PIT. This is different from the Boolean setting, where it is known that strong enough boolean circuit lower bounds imply that $\mathsf{BPP} = \mathsf{P}$. The difference stems from the fact that, in the worst case, an n-variate degree d polynomial P needs to be queried on as many as $\binom{n+d}{d} \gg 2^n$ points to be sure of its non-zeroness. A key player in this interaction of hardness and randomness, in the context of algebraic complexity, is the notion of a *hitting-set generator* (HSG), which we now define.

Definition 1.1 (Hitting-set generators)**.** *A polynomial map $G : \mathbb{F}^k \to \mathbb{F}^m$ given by $G(z_1, z_2, \ldots, z_k) = (g_1(\mathbf{z}), g_2(\mathbf{z}), \ldots, g_n(\mathbf{z}))$ is said to be a hitting-set generator (HSG) for a class $\mathcal{C} \subseteq \mathbb{F}[x_1, x_2, \ldots, x_n]$ of* *polynomials if for every non-zero $Q \in \mathcal{C}$, we have that $Q \circ G = Q(g_1, g_2, \ldots, g_n)$ is also non-zero.*

*We shall say that G is $t(n)$-*explicit *if, for any $\mathbf{a} \in \mathbb{F}^k$ of bit complexity at most n, we can compute $G(\mathbf{a})$ in deterministic time $t(n)$. Here k is called the* seed length *of the HSG and n is called the* stretch *of the HSG. The maximum of the degrees of $g_1, g_2, \ldots, g_n$ is called the* degree *of the HSG.* $\Diamond$

Suppose a polynomial map G is an HSG for a class $\mathcal{C}$ of circuits, we say that the G *fools* the class $\mathcal{C}$ of circuits.

Informally, an HSG G gives a polynomial map which reduces the number of variables in the polynomials in $\mathcal{C}$ from n to k while preserving their non-zeroness. It is not hard to see that such polynomial maps are helpful for deterministic PIT for $\mathcal{C}$. To test if a given n-variate polynomial $Q \in \mathcal{C}$ is non-zero, it is sufficient to check that $Q \circ G$, a k-variate polynomial, is non-zero. If the degree of each g_i is not-too-large, then a "brute-force" check (via the Schwartz-Zippel Lemma) can be used to test if $Q \circ G$ is zero in at most $\mathrm{poly}(t(n)) \cdot (\deg(G) \cdot \deg(Q))^{O(k)}$ time, if G is $t(n)$-explicit. Thus, it is desirable to have HSGs that are very explicit (small $t(n)$), low degree and $\deg(G)$ and large stretch ($k \ll n$).

1.1 Prior construction of generators

Generators from combinatorial designs: One of the earliest (and most well-known) applications of lower bounds to derandomization is the construction of *pseudorandom generator (PRG)* from hard explicit Boolean functions by Nisan and Wigderson [NW94]. In the algebraic setting, an analogous construction was shown to produce HSGs by Kabanets and Impagliazzo [KI04][1]. These constructions are based on the notion of a combinatorial design, which is a family of subsets that have small pairwise intersection. Given an explicit construction of such a combinatorial design (e.g. a family $\mathcal{F} = \{S_1, S_2, \ldots, S_n\}$ of subsets of $[k]$ of size t each), the PRG/HSG in [NW94, KI04] is then constructed by just taking a hard polynomial $P(y_1, \ldots, y_t)$ and defining the map as $G(z_1, z_2, \ldots, z_k) = (P(\mathbf{y}\mid_{S_1}), P(\mathbf{y}\mid_{S_2}), \ldots, P(\mathbf{y}\mid_{S_n}))$. The proof of correctness for this HSG goes via a hybrid argument and a result of Kaltofen [Kal89].

Bootstrapping hitting sets and HSGs with large stretch: In a recent line of work [AGS18, KST19] the following surprising *boostrapping* phenomenon was shown to be true for hitting sets for algebraic circuits. The following is the statement from [KST19]:

Theorem 1.2 ([KST19]). *Let $\delta > 0$ and $n \geq 2$ be constants. Suppose that, for all large enough s, there is an explicit hitting set of size $s^{n-\delta}$ for all degree s, size s algebraic formulas (or algebraic branching programs, or circuits respectively) over n variables. Then, there is an explicit hitting set of size $s^{\exp(\exp(O(\log^* s)))}$ for the class of degree s, size s algebraic formulas (or algebraic branching programs, or circuits respectively) over s variables.*

[1] Even though the construction of the generator in same in [KI04] and [NW94], there are crucial differences in the analysis. In particular, the analysis for the HSG in [KI04] relies on a deep result of Kaltofen [Kal89] about low degree algebraic circuits being closed under polynomial factorization.

In other words, a slightly non-trivial explicit construction of hitting sets even for constant-variate algebraic circuits implies an almost complete derandomization of PIT for algebraic circuits. A natural question in this direction which has remained open is the following.

Question 1.3 ([KST19]). *Can slightly non-trivial hitting sets for constant-variate algebraic circuits can be bootstrapped to get polynomial size (and not just almost polynomial size as in Theorem 1.2) hitting sets for all circuits ?*

The proof of Theorem 1.2 can also be interpreted as a different HSG for algebraic computation. This HSG, given the hypothesis of Theorem 1.2, stretches k bits to n bits (for arbitrarily large n), but the degree and explicitness of the generator grows as $n^{\exp(\exp(O(\log^* n)))}$. Thus, this construction comes very close to answering Question 1.3 without completely answering it.This HSG is essentially constructed via a repeated composition of the HSG in [KI04, NW94], where for each step, it uses a different hard polynomial with an appropriate hardness, which increases gradually. Due to this inherent iterative nature of the construction, it seems difficult to reduce the degree and explicitness of such HSG constructions to $\mathrm{poly}(n)$.

The need to go beyond design-based HSGs: In the set up of boolean computation, observe that we cannot expect to have any PRG (or even HSG) of seed length k to fool circuits of size much larger than $n2^k$ since we can construct a circuit of size $O(n2^k)$ to identify the range of the generator (consisting of 2^k strings of length n each). A similar argument gives an upper bound of $(dD)^{O(k)}$ on the size of degree D algebraic circuits which can be fooled by a HSG with seed length k and degree d. Thus, while the stretch of any boolean PRG constructed via hardness of a boolean function is upper bounded by $n2^k$, in the algebraic setting, one could hope for a construction of hitting set generators of stretch as large as $d^{\Omega(k)}$ from sufficiently hard explicit polynomial families.[2] However, till recently, there were no known constructions of such HSGs with stretch larger than 2^k. An HSG with strong enough parameters would answer the following very natural question.

Question 1.4. *If there is a polynomial family $\{P_k\}_{k\in\mathbb{N}}$, where P_k is an k-variate polynomial of degree d such that any algebraic circuit computing it has size $d^{\Omega(k)}$, then is PIT in* P *?*

Another reason for looking beyond the design based HSGs in the algebraic setting is that by definition, a design-based HSG is combinatorial. Aesthetically, it seems desirable to have a route from algebraic lower bounds to algebraic pseudorandomness which does not rely on clever combinatorial constructions!

PRGs of Shaltiel & Umans [SU05] and Umans [Uma03]: An alternative to the design-based PRGs in the boolean setting is the generator of Shaltiel and Umans [SU05], and a related follow up

[2] Indeed, we know from elementary counting (or dimension counting) arguments that there exist degree d polynomials in k variables which require algebraic circuits of size nearly $\binom{d+k}{k}$, which can be approximated by $d^{\Omega(k)}$ when k is much smaller than d, which is the range of parameters we work with in this paper.

work of Umans [Uma03]. These generators are quite different from the design based generators of Nisan and Wigderson [NW94] and, in particular, appear to be more *algebraic* in their definition and analysis. We refer the interested reader to the original papers [SU05, Uma03] for the formal definitions of these generators and further details.

The algebraic nature of these PRGs makes them good candidates for potential HSGs in the algebraic setting and, indeed, this work was partially motivated by this goal. However, as far as we understand, it remains unclear whether there is an easy adaptation of these PRGs which works for algebraic circuits. In particular, the hardness required for the analysis of the PRGs in [SU05, Uma03] appears to be inherently functional, i.e. they assume that it is hard to evaluate the polynomial over some finite field. In the context of algebraic complexity, the more natural notion of hardness is that it is hard to compute the polynomial syntactically as a formal polynomial via a small algebraic circuit.

1.2 Our Results

Our main result is the construction of a hitting-set generator which comes very close to answering Question 1.4, for characteristic zero fields.

Definition 1.5 (The generator). *For any k-variate polynomial $P(\mathbf{z})$, define the map $\mathsf{Gen}_P : \mathbb{F}^k \times \mathbb{F}^k \to \mathbb{F}^{n+1}$ as follows:*

$$\mathsf{Gen}_P(\mathbf{z}, \mathbf{y}) = (\Delta_0(P)(\mathbf{z}, \mathbf{y}), \Delta_1(P)(\mathbf{z}, \mathbf{y}), \ldots, \Delta_n(P)(\mathbf{z}, \mathbf{y}))\,,$$

where $\Delta_i(P)$ is the homogeneous degree i (in $\mathbf{y}$) component in the Taylor expansion of $P(\mathbf{z} + \mathbf{y})$, i.e.

$$\Delta_i(P)(\mathbf{z}, \mathbf{y}) = \sum_{\mathbf{d} \in \mathbb{N}^k, |\mathbf{e}|_1 = i} \frac{\mathbf{y}^{\mathbf{e}}}{\mathbf{e}!} \cdot \frac{\partial P}{\partial \mathbf{z}^{\mathbf{e}}}. \qquad \Diamond$$

It is clear that the above definition is $d^{O(k)}$-explicit as we can express P as a sum of d^k monomials and compute each component of $\mathsf{Gen}_P(\mathbf{z}, \mathbf{y})$ with a small additional cost. Our main theorem states that the above map is indeed a generator if the polynomial $P(\mathbf{z})$ is hard enough, in the *border* or *infinitesimal approximation* sense. We give an informal definition (over fields such as $\mathbb{C}$ or $\mathbb{R}$) here and this notion shall be discussed in detail in Section 2.2.

Definition (Border computation (informal)). *A polynomial $P \in \mathbb{F}[\mathbf{x}]$ is said to be in the* border *of algebraic circuits of a class $\mathcal{C}$ of algebraic circuits if there is a sequence of size s circuits $\{C_\varepsilon\} \subseteq \mathcal{C}$ (possibly involving coefficients that are rational functions in ε) such that*

$$\lim_{\varepsilon \to 0} C_\varepsilon = P. \qquad \Diamond$$

An example of such a computation is the polynomial $x^{r-1}y$ that is in the border of circuits of the form $\alpha\ell_1^r + \beta\ell_2^r$ where $\alpha, \beta \in \mathbb{F}$ and ℓ_1, ℓ_2 are homogeneous linear polynomials (even though, for any $r \geq 3$, we cannot express $x^{r-1}y$ as $\alpha\ell_1^r + \beta\ell_2^r$).

$$C_\varepsilon := \left(\frac{1}{r\varepsilon}\right) \cdot \left((x+\varepsilon y)^r - x^r\right) \overset{\varepsilon \to 0}{=} x^{r-1}y.$$

Thus, the border of a class of circuits can be more powerful than the class itself. The question of quantitatively understanding this difference in computational power is a fundamental problem, and is of great interest in the context of Geometric Complexity theory.

Our main theorem is the following.

Theorem 1.6 (Main theorem)**.** *Assume that the underlying field $\mathbb{F}$ has characteristic zero. Let P be a polynomial of degree d on k variables such that P is not in the border of algebraic circuits of size at most $\tilde{s}$. Then, for any $(n+1)$-variate polynomial $Q(x_0, \ldots, x_n)$ in the border of algebraic circuits of size s and degree D, if $\left(s \cdot D \cdot d \cdot n^{10k}\right) < \tilde{s}$, then*

$$Q \neq 0 \iff Q \circ \mathsf{Gen}_P(\mathbf{z}, \mathbf{y}) \neq 0.$$

We remark that for our proof, it seems crucial that P is not even in the border of small circuits and is not just hard for small circuits from the point of view of exact computation. Modulo this requirement, Theorem 1.6 almost completely answers Question 1.4 affirmatively. As alluded to in the introduction, we do not know of prior constructions of HSGs with these properties.

In addition to being interesting on its own, Theorem 1.6 leads to the following result which shows that bootstrapping of hitting sets can be done in polynomial time, and at least in the setting of border complexity, answers Question 1.3.

Theorem 1.7 (Bootstrapping in one shot)**.** *Assume that the underlying field $\mathbb{F}$ has characteristic zero. Let $\delta > 0$ be any constant and $k \in \mathbb{N}$ be a large enough constant. Suppose that, for all large enough s, there is an $s^{O(k)}$-explicit hitting set of size $s^{k-\delta}$ for all degree s polynomials which are in the border of size s algebraic circuits over k variables. Then, there is an $s^{O(k^3)}$-explicit hitting set of size $s^{O(k^3)}$ for all of degree s polynomials which are in the border of size s algebraic circuits over s variables.*

Remark. It is worth mentioning that a substantial fraction of lower bounds in algebraic circuit complexity has been proved via *algebraic natural proofs* [GKSS17, FSV18]. Such techniques immediately yield the same lower bounds for border complexity as well.

Also, most of the known constructions of hitting sets for restricted classes of circuits are built by leveraging certain weaknesses exploited in the corresponding lower bound proofs. As a result, many of these hitting sets known for subclasses of algebraic circuits, thus far, are also hitting sets for the border of the respective restricted classes. $\Diamond$

1.3 An overview of the proof

To show that the HSG in Definition 1.5 is indeed a hitting-set generator for low degree polynomials in the border of small circuits, we focus our attention on a purported non-zero polynomial

$Q(\mathbf{x})$ with fewest variables, of border circuit complexity s and degree D which is not *fooled* by the generator, i.e. $Q \circ \mathsf{Gen}_P$ is identically zero. We use this identity to reconstruct a small circuit for P which contradicts its hardness. This would imply that all low degree polynomials in the border of small circuits are fooled by the HSG.

In order to reconstruct a circuit for P from the circuit for Q, we focus on the so called *non-degenerate* case and address it in Lemma 3.2, which is our key technical lemma. Before discussing the main ideas in the proof of Lemma 3.2, we first discuss some of the details of the reduction to the non-degenerate case.

Reducing to the non-degenerate case : In the non-degenerate case we insist that, in addition to having $Q \circ \mathsf{Gen}_P = 0$, we have $(\partial_{x_n} Q) \circ \mathsf{Gen}_P \neq 0$; i.e. the derivative of Q with respect to the *last* variable x_n is *fooled* by the generator. To ensure this condition, we consider the status of the higher order derivatives of the generator with respect to x_n when composed with the generator. Let r be the degree of Q in x_n. If there exists a $j \leq r$ such that $Q, (\partial_{x_n} Q), (\partial_{x_n^2} Q), \ldots, (\partial_{x_n^{j-1}} Q)$ are all non-zero, and vanish when composed with the generator, but $(\partial_{x_n^j} Q) \circ \mathsf{Gen}_P \neq 0$, then, we just work with the the polynomial $\tilde{Q} = (\partial_{x_n^{j-1}} Q)$ instead of Q. Clearly, $\tilde{Q} \circ \mathsf{Gen}_P = 0$ and $\frac{\partial \tilde{Q}}{\partial x_n} \circ \mathsf{Gen}_P \neq 0$, and we are in the case handled by Lemma 3.2. Moreover, the complexity of $\tilde{Q}$ is not much larger than that of Q; more precisely, it follows by a simple interpolation argument that $\tilde{Q}$ is in the border of circuits of size at most $O(sD)$. We invoke Lemma 3.2 now with these parameters, and that would complete the proof.

We still need to consider the case that there is no such $j \leq r$ such that $(\partial_{x_n^j} Q) \circ \mathsf{Gen}_P \neq 0$, in particular, $(\partial_{x_n^r} Q) \circ \mathsf{Gen}_P = 0$. Since r equals the degree of Q in x_n, it follows that $\tilde{Q} = (\partial_{x_n^r} Q)$ is a polynomial on one fewer variable than Q which is non-zero and vanishes when composed with the generator. This can be handled by assuming that Q was the *minimal* (in terms of the number of variables it depends on) non-zero, degree $\leq D$ polynomial in the border of size s circuits that is *not* fooled by our generator.

> **Hurdle:** The circuit complexity of $\partial_{x_n^r} Q$ is, typically, a little larger than the complexity of Q. Even if there is a slight increase in size, how does $(\partial_{x_n^r} Q) \circ \mathsf{Gen}_P = 0$ contradict minimality of Q?

This is the one of the key places where get help from the border. The crucial observation is that although we do not know if $(\partial_{x_n^r} Q)$ is *computable* by a circuit of size at most s, we show in Lemma 2.8 that the border complexity of $(\partial_{x_n^r} Q)$ is upper bounded by the border complexity of Q. *This* would then be enough to leverage the minimality assumption on Q.

The proof of Lemma 2.8 is a simple trick with border computation, and is a slight variant of the *dampening* trick often used in this context (e.g. see [Kum18]).

The proof of Lemma 3.2 : The proof of the lemma can be viewed as a variant of the standard Newton Iteration (or Hensel lifting) based argument often used in the context of root finding, although there are some crucial differences. We iteratively construct the polynomial $P(\mathbf{z})$ one homogeneous component at a time (recall that $P(\mathbf{z})$ is a k-variate polynomial of degree d). In fact, our induction hypothesis needs to be a bit stronger than this. For our proof, we maintain the invariant that at the end of the i^{th} iteration, we have a multi-output circuit which ε computes all the partial derivatives of order at most n of all the homogeneous components of $P(\mathbf{z})$ of degree at most $i+n$. However, for this overview, we ignore this technicality and pretend that we are directly working with the homogeneous components of $P(\mathbf{z})$. For the base case, we assume that we have access to all the homogeneous components of P of degree at most n, which are homogeneous polynomials of degree at most n on k variables and are trivially computable by a circuit of size at most $n^{O(k)}$, which is much smaller than $d^{\Omega(k)}$, the presumed hardness of P for $d \gg n$. Thus, we have n homogeneous components of $P(\mathbf{z})$, and the goal is to use them and the non-degeneracy assumption to reconstruct all of P. Let us assume that we have already computed $P_0, \ldots, P_i$, where P_j is the homogeneous component of $P(\mathbf{z})$ of degree equal to j. We now focus on recovering the homogeneous component P_{i+1} of degree equal to $i+1$. Observe that $\Delta_n(P_{i+1})$ is a homogeneous (in $\mathbf{z}$) polynomial of degree $(i-n+1)$. We show that given the non-degeneracy condition in the hypothesis of the lemma, there is a small circuit such that modulo the ideal $\langle \mathbf{z} \rangle^{i-n+2}$, it computes $\Delta_n(P_{i+1})(\mathbf{z}, \mathbf{y})$. Since $\Delta_n(P_{i+1})(\mathbf{z}, \mathbf{y})$ is essentially a *generic* linear combination of n-th order derivatives of $P_{i+1}(\mathbf{z})$, it is not hard to show[3] that we can obtain a small circuit that outputs each of the n-th order partial derivatives of $P_{i+1}(\mathbf{z})$, modulo higher degree monomials. Then we would be able to reconstruct $P_{i+1}(\mathbf{z}) \bmod \langle \mathbf{z} \rangle^{i+2}$ via repeated applications of the Euler's differentiation formula for homogeneous polynomials.

Fact 1.8 (Euler's formula for differentiation of homogeneous polynomials)**.** *If $A(x_1, \ldots, x_k)$ is a homogeneous polynomial of degree t, then $\sum_{i=1}^{k} \partial_{x_i} A = t \cdot A(x_1, \ldots, x_k)$.*

One crucial point in this entire reconstruction is that each step of the reconstruction only incurs an *additive* blow-up in size and hence can be repeated for polynomially many steps to recover each homogeneous part of P (Figure 1 in Section 3 contains a pictorial description of the inductive step).

> **Hurdle:** This still only gives a small circuit which modulo the ideal $\langle \mathbf{z} \rangle^{i+2}$ computes $P_{i+1}(\mathbf{z})$ and hence we need to extract the homogeneous part of degree $(i+1)$. Typically, extracting a certain homogeneous part requires an interpolation step and this incurs a multiplicative blow-up in size which is unaffordable in this setting.

Once again, the setting of border complexity is crucial. As shown in Lemma 2.9, if Q is in the border of size s circuits, then the lowest (or highest) degree homogeneous part of Q is also in the border of size s circuits (this is again proved via a similar dampening trick). Thus, in the setting of border complexity, extracting extremal homogeneous components incurs *no* cost at all! We also remark that in the usual Newton Iteration based argument for computing roots of polynomials, this homogenization is not necessary in every step, but it seems necessary for our proof.

[3]In particular, this part of the argument relies on the stronger hypothesis that we have access to each of the order n partial derivatives of $P_0, P_1, \ldots, P_i$.

Overall this merely additive increase in size allows us to run the reconstruction step to extract all homogeneous components of P and showing P is in the border of small circuits, contradicting the hardness of P.

Similarities with [SU05, Uma03] and [Kop15] : We remark that at a high level, our construction of the HSG was inspired by the constructions by Shaltiel and Umans [SU05, Uma03], although the precise form of our generator seems different from that in [SU05, Uma03]. We also note that the set up of induction we have in the proof of Lemma 3.2 is very similar to the set up used by Kopparty [Kop15] in the context of list decoding Multiplicity codes. More precisely, our induction is similar to what is used in constructing a power series expansion of a non-degenerate solution of the univariate Cauchy-Kovalevski differential equations, which are used in [Kop15]. The key difference is that while we work with a multivariate setting, the iterative proof of Lemma 3.2 resembles the list decoding algorithm for univariate multiplicity codes in [Kop15]. It appears to be of interest to understand this analogy further.

2 Notation and preliminaries

- Throughout the paper, we think of $\mathbb{F}$ as a field of characteristic zero (or large enough).
- We use boldface letters such as $\mathbf{z}$ to denote sets or tuples: $\mathbf{z} = (z_1, z_2, \ldots, z_k)$. For an exponent vector $\mathbf{e}$, we shall use $\mathbf{z}^{\mathbf{e}}$ to denote the monomial $z_1^{e_1} \cdots z_k^{e_k}$. Let $|\mathbf{e}| := \sum e_i$.
- We use $\partial_{\mathbf{z}^{\mathbf{e}}}(P(\mathbf{z}))$ to denote the partial derivative $\frac{\partial^{|\mathbf{e}|}(P)}{\partial \mathbf{z}^{\mathbf{e}}}$.
- We use $\langle \mathbf{z} \rangle^i$ to denote the ideal in $\mathbb{F}[\mathbf{z}]$ generated by all degree i monomials in $\mathbf{z}$.
- We use $\mathcal{P}(k, d)$ to denote the class of k-variate polynomials of degree at most d.

2.1 PIT preliminaries

The following well-known lemma gives an exponential (in the number of variables) sized hitting set for the class of degree d polynomials.

Lemma 2.1 ([Ore22, DL78, Sch80, Zip79])**.** *Let $f(\mathbf{x})$ be a non-zero n-variate polynomial of degree at most d. Then for any set $S \subset \mathbb{F}$ with $|S| > d$, there is a point $\mathbf{a} \in S^{|\mathbf{x}|}$ such that $f(\mathbf{a}) \neq 0$.* $\square$

It is also known that existence of non-trivial hitting sets for a class $\mathcal{C}$ can be used to construct hard polynomials.

Theorem 2.2 (Informal, Heintz and Schnorr [HS80], Agrawal [Agr05])**.** *Let $H(n, d, s)$ be an explicit hitting set for circuits of size s, degree d in n variables. Then, for every $k \leq n$ and d' such that $d'k \leq d$ and $(d'+1)^k > |H(n, d, s)|$, there is a non-zero polynomial on n variables and individual degree d' that vanishes on the hitting set $H(n, d, s)$, and hence cannot be computed by a circuit of size s.*

Finally, we need the following notion of *interpolating sets* for a class of polynomials.

Definition 2.3 (Interpolating sets for $\mathcal{P}(k,d)$)**.** *Let $M_{k,d}$ denote the number of k-variate monomials of degree at most d. That is, $M_{k,d} = \binom{k+d}{d}$.*

A set of points $\mathbf{a}_1, \ldots, \mathbf{a}_r \in \mathbb{F}^k$ is said to be an interpolating set for $\mathcal{P}(k,d)$ *if the vectors*

$$\left\{ \left(\mathbf{a}_i^{\mathbf{e}} \;:\; \mathbf{e} \in \mathbb{Z}_{\geq 0}^k \,,\; |\mathbf{e}| \leq d \right) \;:\; i \in [r] \right\} \subset \mathbb{F}^{M_{k,d}}$$

form a spanning set for $\mathbb{F}^{M_{k,d}}$.

Equivalently, there exists field constants $\beta_1, \ldots, \beta_r$ such that for every $f(\mathbf{z}) \in \mathcal{P}(k,d)$ and every $\mathbf{e} \in \mathbb{Z}_{\geq 0}^k$ with $|\mathbf{e}| \leq d$, we have that

$$\operatorname{coeff}_{\mathbf{z}^{\mathbf{e}}}(f) = \sum_{i=1}^{r} \beta_i f(\mathbf{a}_i). \qquad \Diamond$$

A canonical example of an interpolating set for $\mathcal{P}(k,d)$ is $S^k = \{(s_1, \ldots, s_k) \;:\; s_i \in S \;\forall i\}$ where $S \subseteq \mathbb{F}$ is a set of at least $(d+1)$ distinct field elements. The following well-known proposition says that a random set of points, of the appropriate size, is an interpolating set for $\mathcal{P}(k,d)$ with high probability if the field $\mathbb{F}$ is large enough.

Proposition 2.4 (Random sets are interpolating sets)**.** *For any d, k, if $\mathbb{F}$ is large enough, then a random set of size $\binom{k+d}{d}$ is an interpolating set for $\mathcal{P}(k,d)$ with probability $1 - o(1)$.* $\square$

2.2 Border computation

Definition 2.5 (ε-computing a function)**.** *A circuit C over $\mathbb{F}(\varepsilon)[\mathbf{x}]$ is said to ε-compute a polynomial $Q(\mathbf{x})$, denoted by $C =_\varepsilon Q$, if the output of the circuit C is a polynomial in $\mathbb{F}[\mathbf{x}, \varepsilon]$ such that*

$$C(\mathbf{x};\varepsilon) = Q(\mathbf{x}) + \varepsilon \cdot C'(\mathbf{x};\varepsilon).$$

for some polynomial $C'(\mathbf{x}, \varepsilon) \in \mathbb{F}[\mathbf{x}, \varepsilon]$. In particular, $\lim_{\varepsilon \to 0} C(\mathbf{x};\varepsilon) = Q(\mathbf{x})$.

(We use the notation $C(\mathbf{x};\varepsilon)$ as opposed to $C(\mathbf{x},\varepsilon)$ to distinguish between the actual variables of the circuit $\mathbf{x}$ and the convergence parameter ε.) $\Diamond$

In other words, $C =_\varepsilon Q$ implies that setting $\varepsilon = 0$ *in the output of* C results in Q (though the circuit C could involve internal constants with ε's in the denominators). As mentioned earlier, the following is an example:

$$C := \left(\frac{1}{r\varepsilon}\right) \cdot \left((x + \varepsilon y)^r - x^r\right)$$

is a circuit that ε-computes the polynomial $x^{r-1}y$.

In other words, if we let $\mathbb{F}$ be the field of complex numbers and think of ε as a constant tending to zero, then in some sense, the circuit C in the definition approximates the polynomial P up to an error ε. As ε tends to zero the magnitude of the constants in the circuit C tends to infinity (while its size remains the same), and we get closer and closer to P. The notion of border complexity plays a key role in connecting questions in algebraic complexity to underlying questions in algebra and geometry. In particular, understanding whether going to the border of a complexity class of polynomials endows it with additional computational power is a natural and fundamental question in Geometric Complexity theory. For a more detailed discussion on border complexity we refer the interested reader to [BIZ18] and references therein.

Composition is well behaved for border computation

Lemma 2.6. *Let $Q \in \mathbb{F}[\mathbf{x}, y]$ and $P \in \mathbb{F}[\mathbf{x}]$ be two polynomials which is in the border of algebraic circuits of size s_1 and s_2 respectively. Then, $Q(\mathbf{x}, P)$ is in the border of algebraic circuits of size $s_1 + s_2$.*

Proof. Let $C \in \mathbb{F}(\varepsilon)[\mathbf{x}, y]$ be a circuit of size at most s_1 which approximates Q. In other words, there are polynomials $A_1, A_2, \ldots, A_t \in \mathbb{F}[\mathbf{x}, y]$ such that

$$C(\mathbf{x}, y) \equiv Q + \sum_{i=1}^{t} \varepsilon^i A_i \, .$$

Similarly, let $\Phi \in \mathbb{F}(\varepsilon)[\mathbf{x}]$ be a circuit of size at most s_2 which approximates P. In other words, there are polynomials $B_1, B_2, \ldots, B_r \in \mathbb{F}[\mathbf{x}]$ such that

$$\Phi(\mathbf{x}) \equiv P + \sum_{j=1}^{r} \varepsilon^j B_j \, .$$

We now prove the natural and intuitive claim that

$$\lim_{\varepsilon \to 0} C(\mathbf{x}, \Phi(\mathbf{x})) = Q(\mathbf{x}, P) \, .$$

This would complete the proof of the lemma.

$$\begin{aligned} C(\mathbf{x}, \Phi(\mathbf{x})) &= Q(\mathbf{x}, \Phi(\mathbf{x})) + \sum_{i=1}^{t} \varepsilon^i A_i(\mathbf{x}, \Phi(\mathbf{x})) \\ &= Q\left(\mathbf{x}, P + \sum_{j=1}^{r} \varepsilon^j B_j\right) + \sum_{i=1}^{t} \varepsilon^i A_i \left(\mathbf{x}, P + \sum_{j=1}^{r} \varepsilon^j B_j\right) \\ &= Q\left(\mathbf{x}, P\right) + \sum_{j=1}^{t'} \varepsilon^j \cdot A'_j(\mathbf{x}) + \sum_{i=1}^{t} \varepsilon^i A_i \left(\mathbf{x}, P + \sum_{j=1}^{r} \varepsilon^j B_j\right) \, , \end{aligned}$$

where the last step follows from a Taylor expansion of $Q\left(\mathbf{x}, P + \sum_{j=1}^{r} \varepsilon^j B_j\right)$ around the point $(\mathbf{x}, P)$. $\square$

Iterative application of the lemma gives the following corollary.

Corollary 2.7. *Let $Q \in \mathbb{F}[\mathbf{x}, y_1, y_2, \ldots, y_n]$ be a polynomial in the border of circuits of size s_0 and $P_1, P_2, \ldots, P_n \in \mathbb{F}[\mathbf{x}]$ be polynomials which are in the border of algebraic circuits of size $s_1, s_2, \ldots, s_n$ respectively. Then, $Q(\mathbf{x}, P_1, P_2, \ldots, P_n)$ is in the border of algebraic circuits of size $s_0 + s_1 + s_2 + \ldots + s_n$.*

Extremal derivatives and homogeneous parts

Lemma 2.8. *Let $Q \in \mathbb{F}[\mathbf{x}, y]$ be a polynomial of degree equal to d in y, and let $P \in \mathbb{F}[\mathbf{x}, y, \varepsilon]$ be a polynomial which can be computed by a circuit $C \in \mathbb{F}(\varepsilon)[\mathbf{x}, y]$ of size s, such that*

$$\lim_{\varepsilon \to 0} P = Q .$$

Then, $\frac{\partial Q}{\partial y^d}$ is also in the border of algebraic circuits of size at most s.

Proof. We may assume that $d > 0$ (for otherwise, there is nothing to prove). Let $D - 1$ be the degree of P in y. Then,

$$\begin{aligned} P(\mathbf{x}, y; \varepsilon) &= Q(\mathbf{x}, y) + \varepsilon \cdot \tilde{P}(\mathbf{x}, y; \varepsilon) \\ \implies P(\mathbf{x}, y; \varepsilon^D) &= Q(\mathbf{x}, y) + \varepsilon^D \cdot \tilde{P}(\mathbf{x}, y; \varepsilon^D) \\ \implies \varepsilon^d \cdot P(\mathbf{x}, (y/\varepsilon); \varepsilon^D) &= \varepsilon^d Q(\mathbf{x}, (y/\varepsilon)) + \varepsilon^{D+d} \cdot \tilde{P}(\mathbf{x}, (y/\varepsilon); \varepsilon^D) \end{aligned}$$

Since $\tilde{P}(\mathbf{x}, y; \varepsilon)$ is an honest-to-god polynomial in $\mathbf{x}, y$ and ε and hence so is $\varepsilon \cdot \tilde{P}(\mathbf{x}, y; \varepsilon)$ with each coefficient being divisible by ε. Hence, $\varepsilon^D \cdot \tilde{P}(\mathbf{x}, y; \varepsilon^D)$ has each coefficient divisible by ε^D. As the degree of $\tilde{P}$ in y is at most $D - 1$, we have $\varepsilon^D \cdot \tilde{P}(\mathbf{x}, (y/\varepsilon); \varepsilon^D)$ is also a polynomial in $\mathbf{x}, y$ and ε with each coefficient being divisible by ε. Finally, since $d > 0$, we have that each coefficient of the polynomial $\varepsilon^{D+d} \cdot \tilde{P}(\mathbf{x}, (y/\varepsilon); \varepsilon^D)$ is divisible by ε. Hence,

$$\begin{aligned} \lim_{\varepsilon \to 0} \left(\varepsilon^d \cdot P(\mathbf{x}, (y/\varepsilon); \varepsilon^D) \right) &= \lim_{\varepsilon \to 0} \left(\varepsilon^d \cdot Q(x, (y/\varepsilon)) \right) \\ &= \frac{\partial^d Q}{\partial y^d} \cdot y^d. \end{aligned}$$

Thus, by setting $y = 1$, this immediately yields a circuit of size at most s that approximates $\frac{\partial^d Q}{\partial y^d}$. ☐

A very similar argument also gives the following lemma which would also be useful for us.

Lemma 2.9 (Extracting the lowest-degree homogeneous parts)**.** *Let $P_1, \ldots, P_m \in \mathbb{F}[\mathbf{x}]$ and suppose $P_i = Q_i + R_i$ where Q_i is the lowest-degree homogeneous part of P_i. Given a multi-output circuit $C(\mathbf{x}; \varepsilon)$ of size s that ε-computes $\{P_1, \ldots, P_m\}$. Then, $\{Q_1, \ldots, Q_m\}$ can also be ε-computed by a multi-output circuit $\tilde{C}$ of size s.*

Proof. The proof is exactly along the lines as Lemma 2.8. Suppose the outputs of $C(\mathbf{x}; \varepsilon)$ are

$$\tilde{P}_1(\mathbf{x}; \varepsilon) = (Q_1(\mathbf{x}) + R_1(\mathbf{x})) + \varepsilon \cdot S_1(\mathbf{x}; \varepsilon)$$

$$\vdots$$

$$\tilde{P_m}(\mathbf{x};\varepsilon) = (Q_1(\mathbf{x}) + R_1(\mathbf{x})) + \varepsilon \cdot S_m(\mathbf{x};\varepsilon)$$

Let $d_i = \deg(\tilde{P}_i)$ and $D = \max(\{d_i\}_i) + 1$. As in Lemma 2.8, the circuit $C(\varepsilon x_1, \ldots, \varepsilon x_n; \varepsilon^D)$ has outputs

$$\begin{aligned}
\hat{P}_i(\mathbf{x};\varepsilon) &= Q_i(\varepsilon\mathbf{x}) + R_i(\varepsilon\mathbf{x}) + \varepsilon^D S_i(\varepsilon\mathbf{x};\varepsilon^D) \\
&= \varepsilon^{d_i} Q_i(\mathbf{x}) + R_i(\varepsilon\mathbf{x}) + \varepsilon^D S_i(\varepsilon\mathbf{x};\varepsilon^D) \\
&= \varepsilon^{d_i} Q_i(\mathbf{x}) \bmod \varepsilon^{d_i+1}
\end{aligned}$$

for each i. By rescaling the i-th output by ε^{-d_i}, we have a circuit that ε-computes $Q_1, \ldots, Q_m$. $\square$

2.3 The Generator

For a k-variate polynomial P, let $\Delta_i(P)(\mathbf{z}, \mathbf{y}) \in \mathbb{F}[\mathbf{z}, \mathbf{y}]$ defined as

$$\Delta_i(P) = \sum_{\mathbf{e}:|\mathbf{e}|=n} \left(\frac{\mathbf{y}^{\mathbf{e}}}{\mathbf{e}!}\right) \cdot \partial_{\mathbf{z}^{\mathbf{e}}}(P)$$

where $\mathbf{e}! = e_1! \cdots e_k!$. The generator with respect to P is defined as follows:

$$\mathsf{Gen}_P(\mathbf{z}, \mathbf{y}) = (\Delta_0(P)(\mathbf{z}, \mathbf{y}), \ldots, \Delta_n(P)(\mathbf{z}, \mathbf{y}))\,.$$

The following is a simple observation about the operator Δ.

Observation 2.10. *Let $P(\mathbf{z})$ and $Q(\mathbf{z})$ be polynomials such that $P = Q \bmod \langle \mathbf{z} \rangle^j$. Then, for any $i \leq j$, we have $\Delta_i(P) = \Delta_i(Q) \bmod \langle \mathbf{z} \rangle^{j-i}$.*

3 The Main Theorem

We start by recalling the main theorem.

Theorem 1.6 (Main theorem)**.** *Assume that the underlying field $\mathbb{F}$ has characteristic zero. Let P be a polynomial of degree d on k variables such that P is not in the border of algebraic circuits of size at most $\tilde{s}$. Then, for any $(n+1)$-variate polynomial $Q(x_0, \ldots, x_n)$ in the border of algebraic circuits of size s and degree D, if $\left(s \cdot D \cdot d \cdot n^{10k}\right) < \tilde{s}$, then*

$$Q \neq 0 \iff Q \circ \mathsf{Gen}_P(\mathbf{z}, \mathbf{y}) \neq 0.$$

The rest of this section would be devoted to the proof of this theorem.

Let us assume the contrary. That is, there is a circuit $C(\mathbf{x};\varepsilon)$ of size s and degree D such that $Q = \lim_{\varepsilon\to 0} C \neq 0$ but $Q \circ \mathsf{Gen}_P(\mathbf{z},\mathbf{y}) = \lim_{\varepsilon\to 0}(C \circ \mathsf{Gen}_P(\mathbf{z},\mathbf{y})) = 0$. We shall assume, without loss of generality, that $\lim_{\varepsilon\to 0} C$ depends non-trivially on the variable x_n and that no circuit $C'(\mathbf{x};\varepsilon)$ of size s and degree D with $\lim_{\varepsilon\to 0} C'$ depending on fewer variables satisfy $\lim_{\varepsilon\to 0} C' \neq 0$ and $\lim_{\varepsilon\to 0} C' \circ \mathsf{Gen}_P(\mathbf{z},\mathbf{y}) = 0$.

The proof will proceed by inductively building a circuit that ε-computes each homogeneous part of P but we would need the following preprocessing.

Preprocessing the circuit: Let $C(x_0,\ldots,x_n;\varepsilon)$ be the minimal (in terms of number of variables) size s circuit that is *not* fooled by $\mathsf{Gen}_P(\mathbf{z},\mathbf{y})$. That is, $C \circ \mathsf{Gen}_P(\mathbf{z},\mathbf{y}) =_\varepsilon 0$.

Claim 3.1. *There is some $i \geq 0$ such that*

$$\begin{aligned}
\partial_{x_n^i}(C) \circ \mathsf{Gen}_P(\mathbf{z},\mathbf{y}) &=_\varepsilon 0,\\
\partial_{x_n^{i+1}}(C) \circ \mathsf{Gen}_P(\mathbf{z},\mathbf{y}) &\neq_\varepsilon 0.
\end{aligned}$$

Proof. Let $r = \deg_{x_n}(\lim_{\varepsilon\to 0} C)$. Then, the polynomial $0 \neq Q' = \partial_{x_n^r}\left(\lim_{\varepsilon\to 0} C\right)$ does not depend on x_n. Furthermore, by Lemma 2.8, we know that Q' can also be ε-computed by circuits of size s and degree D. Thus, by the minimality of the choice of C, we have that

$$0 \neq_\varepsilon Q' \circ \mathsf{Gen}_P(\mathbf{z},\mathbf{y}) =_\varepsilon \partial_{x_n^r}(C) \circ \mathsf{Gen}_P(\mathbf{z},\mathbf{y}).$$

Since $C \circ \mathsf{Gen}_P(\mathbf{z},\mathbf{y}) =_\varepsilon 0$ and $\partial_{x_n^r}(C) \circ \mathsf{Gen}_P(\mathbf{z},\mathbf{y}) \neq_\varepsilon 0$, there must be an intermediate derivative where a switch from zero to non-zero occurs. $\square$

Let $C' = \partial_{x_n^i}(C)$. In what follows, we will work with C′ instead of C. Let its size be $s' \leq s \cdot D$ (where $D = \deg(C)$).

$$\begin{aligned}
C' \circ \mathsf{Gen}_P(\mathbf{z},\mathbf{y}) &=_\varepsilon 0,\\
\partial_{x_n}(C') \circ \mathsf{Gen}_P(\mathbf{z},\mathbf{y}) &\neq_\varepsilon 0.
\end{aligned}$$

Without loss of generality (by translating $\mathbf{z}$ if necessary), assume that

$$\left(\partial_{x_n}(C') \circ \mathsf{Gen}_P(\mathbf{0},\mathbf{y})\right) \;=\; \Psi(\mathbf{y};\varepsilon) \text{ and } \lim_{\varepsilon\to 0}\Psi(\mathbf{y};\varepsilon) = \Psi(\mathbf{y}) \neq 0.$$

Let $P = P_0 + P_1 + \cdots + P_d$ be the decomposition into homogeneous parts, with P_i being the homogeneous part of degree i, and let $P_{\leq r} := \sum_{i \leq r} P_i$.

Base case $(j = 0)$ **:** Each $\partial_{\mathbf{z}^\mathbf{e}} P_\ell$ for $|\mathbf{e}| \leq n$ and $\ell \leq n$ can be explicitly written as a sum of at most $N := \binom{n+k}{k}$ monomials. Hence, there is a circuit B_0 of size $s_0 = N^2$ that ε-computes (in fact, even exactly computes) $\{\partial_{\mathbf{z}^\mathbf{e}}(P_\ell) \;:\; 0 \leq \ell \leq n\,,\; |\mathbf{e}| \leq n\}$.

Induction hypothesis: There is a circuit $B_{j-1}(\mathbf{z};\varepsilon)$ of size at most s_{j-1}, with $N(n+j-1)$ outputs that ε-computes $\partial_{\mathbf{z}^{\mathbf{e}}} P_\ell$ for each $\mathbf{e}$ with $|\mathbf{e}| \leq n$ and $\ell \leq n+j-1$.

Induction step: To construct a circuit $B_j(\mathbf{z};\varepsilon)$ of size at most s_j (to be defined shortly) that ε-computes $\partial_{\mathbf{z}^{\mathbf{e}}} P_\ell$ for each $\mathbf{e}$ with $|\mathbf{e}| \leq n$ and $\ell \leq n+j$.

Recall $N = \binom{n+k}{n}$, the number of k-variate degree n monomials. We shall say that $\mathbf{a} \in \mathbb{F}^n$ is "good" if $\Psi(\mathbf{a}) \neq 0$. Since $\mathbb{F}$ is large enough, by Proposition 2.4 and Lemma 2.1, a random set $\{\mathbf{a}_1, \ldots, \mathbf{a}_N\} \subset \mathbb{F}^n$ is a set of "good" points that is also an interpolating set for $\mathcal{P}(k,n)$ with probability $1 - o(1)$. Let $\Gamma_{j-1,\mathbf{a}}$ be defined as

$$\Gamma_{j-1,\mathbf{a}} := (\Delta_0(P_{\leq n+j-1})(\mathbf{z},\mathbf{a}), \ldots, \Delta_n(P_{\leq n+j-1})(\mathbf{z},\mathbf{a})).$$

Lemma 3.2. *Let $\mathbf{a} \in \mathbb{F}^k$ be such that $0 \neq \Psi(\mathbf{a}) = \lim_{\varepsilon \to 0} ((\partial_{x_n} C') \circ \mathsf{Gen}_P(\mathbf{0},\mathbf{a}))$. Then,*

$$\left(\frac{-1}{\Psi(\mathbf{a})}\right) \cdot C'(\Gamma_{j-1,\mathbf{a}}) =_\varepsilon \Delta_n(P_{n+j})(\mathbf{z},\mathbf{a}) \bmod \langle \mathbf{z} \rangle^{j+1}.$$

We will defer the proof of this lemma to the end of the section and finish the rest of the proof.

We can begin with the circuit $B_{j-1}(\mathbf{z};\varepsilon)$ that ε-computes every $\partial_{\mathbf{z}^{\mathbf{e}}}(P_\ell)$ for $|\mathbf{e}| \leq n$ and $\ell \leq n+j-1$. By taking suitable linear combinations of the output gates, we can create a new circuit B, of size at most $s_{j-1} + N^5$, that ε-computes $\left\{\Gamma_{j-1,\mathbf{a}_t} \;:\; t \in [N]\right\}$. Using Lemma 3.2 for each $\mathbf{a}_i$, we then obtain a circuit of size $s_{j-1} + N^5 + s' \cdot N$ that ε-computes $\left\{\Delta_n(P_{n+j})(\mathbf{z},\mathbf{a}_t) \;:\; t \in [N]\right\}$ modulo the ideal $\langle \mathbf{z} \rangle^{j+1}$.

By definition, $\Delta_n(P_{n+j})(\mathbf{z},\mathbf{a})$ is a suitable linear combination of the n-th order partial derivatives of $P_{n+j}(\mathbf{z})$. Since $\{\mathbf{a}_1, \ldots, \mathbf{a}_N\}$ was chosen to be an interpolating set, each $\partial_{\mathbf{z}^{\mathbf{e}}}(P_{n+j})$ with $|\mathbf{e}| = n$ can be written as a linear combination of $\left\{\Delta_n(P_{n+j})(\mathbf{z},\mathbf{a}_t) \;:\; i \in [N]\right\}$. As $\{\mathbf{a}_1, \ldots, \mathbf{a}_N\}$ was chosen to be an interpolating set, each $\partial_{\mathbf{z}^{\mathbf{e}}}(P_{n+j})$ with $|\mathbf{e}| = n$ can be written as a suitable linear combination of $\left\{\Delta_n(P_{n+j})(\mathbf{z},\mathbf{a}_t) \;:\; t \in [N]\right\}$. Furthermore, since P_{n+j} is a homogeneous polynomial, we can also compute all its lower order derivatives via repeated applications of Euler's formula (Fact 1.8). Overall, combined with the outputs of $B_{j-1}(\mathbf{z};\varepsilon)$, we have a circuit $B'_j(\mathbf{z};\varepsilon)$ (shown in Figure 1) of size $s_{j-1} + N^{10} + s'N$ that ε-computes

$$\left\{\partial_{\mathbf{z}^{\mathbf{e}}}(P_\ell) \;:\; |\mathbf{e}| \leq n\,,\; \ell \leq n+j-1\right\} \cup \left\{\partial_{\mathbf{z}^{\mathbf{e}}}(\tilde{P}_{n+j}) \;:\; |\mathbf{e}| \leq n\right\},$$

where $\partial_{\mathbf{z}^{\mathbf{e}}}(\tilde{P}_{n+j}) \mod \langle \mathbf{z} \rangle^{n+j-|\mathbf{e}|+1} \equiv \partial_{\mathbf{z}^{\mathbf{e}}}(P_{n+j})$ for every $|\mathbf{e}| \leq n$. Using Lemma 2.9, extracting the lowest degree homogeneous components of these outputs, gives a circuit B_j of size $s_j \leq s_{j-1} + N^{10} + s'N$ that ε-computes

$$\left\{\partial_{\mathbf{z}^{\mathbf{e}}}(P_\ell) \;:\; |\mathbf{e}| \leq n\,,\; \ell \leq n+j\right\}.$$

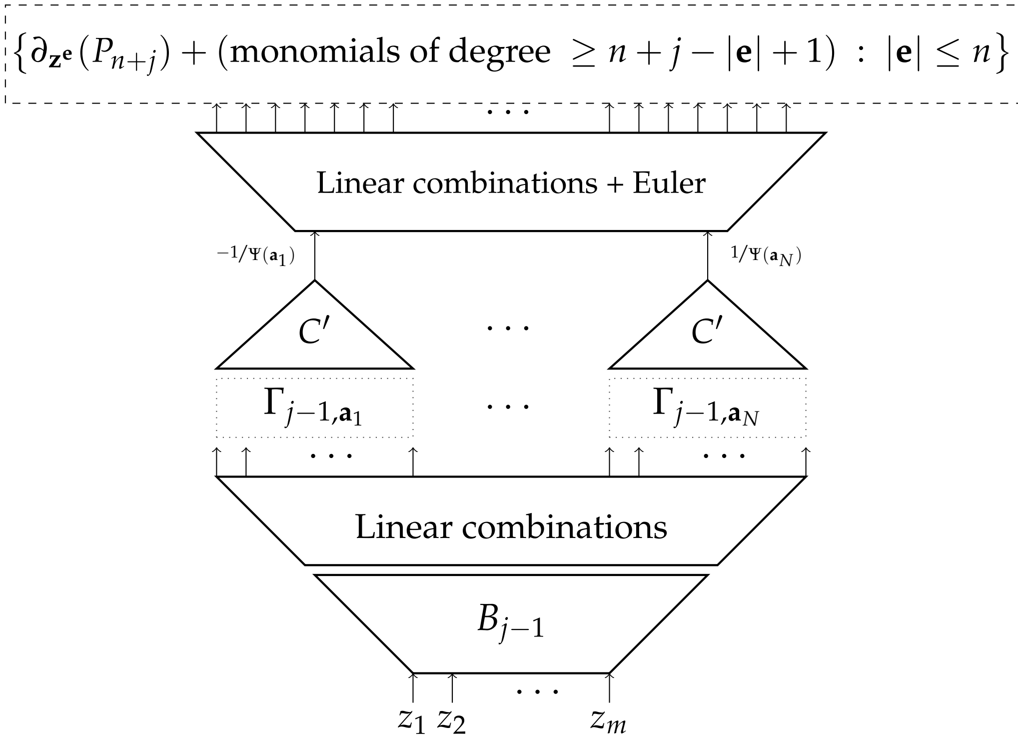

Figure 1: Pictorial representation of B'_j

This completes the induction step.

Unraveling the induction for $d-n$ steps, we eventually obtain a circuit of size at most $s' \cdot d \cdot N^{10} = s \cdot D \cdot d \cdot N^{10}$ that ε-approximates all the partial derivatives of order at most n of $P_0, \ldots, P_d$, and thus from Fact 1.8, we can recover P. However, this contradicts the hardness assumption of P. Hence, it must be the case that $\lim_{\varepsilon \to 0} C \circ \mathsf{Gen}_P(\mathbf{z}, \mathbf{y}) \neq 0$. This completes the proof of the main theorem barring the proof of Lemma 3.2; we address this next. $\square$(Theorem 1.6)

3.1 Proof of Lemma 3.2

We are given $\Gamma_{j-1,\mathbf{a}} = (\Delta_0(P_{\leq n+j-1})(\mathbf{z}, \mathbf{a}), \ldots, \Delta_n(P_{\leq n+j-1})(\mathbf{z}, \mathbf{a}))$. From the assumption on C', we have

$$\begin{aligned} & 0 =_\varepsilon C'(\Delta_0(P)(\mathbf{z}, \mathbf{a}), \ldots, \Delta_n(P)(\mathbf{z}, \mathbf{a}); \varepsilon) \\ \implies & 0 =_\varepsilon C'(\Delta_0(P)(\mathbf{z}, \mathbf{a}), \ldots, \Delta_n(P)(\mathbf{z}, \mathbf{a}); \varepsilon) \bmod \langle \mathbf{z} \rangle^{j+1} \end{aligned}$$

By Observation 2.10, we have that $\Delta_i(P)(\mathbf{z}, \mathbf{a}) = \Delta_i(P_{\leq n+j-1})(\mathbf{z}, \mathbf{a}) \bmod \langle \mathbf{z} \rangle^{j+1}$ for all $i \leq n-1$, and $\Delta_n(P)(\mathbf{z}, \mathbf{a}) = \Delta_n(P_{\leq n+j-1})(\mathbf{z}, \mathbf{a}) + \Delta_n(P_{n+j})(\mathbf{z}, \mathbf{a}) \bmod \langle \mathbf{z} \rangle^{j+1}$. For the sake of brevity, let $R_i =$

$\Delta_i(P_{\leq n+j-1})(\mathbf{z}, \mathbf{a})$ for $0 \leq i \leq n$ and $A = \Delta_n(P_{n+j})(\mathbf{z}, \mathbf{a})$. Therefore,

$$0 =_\varepsilon C'(R_0, R_1, \ldots, R_{n-1}, R_n + A; \varepsilon) \bmod \langle \mathbf{z} \rangle^{j+1}.$$

We now expand the above expression as a *univariate in A* (or in other words, perform a Taylor expansion of the polynomial C' around the point $(R_0, R_1, \ldots, R_n)$).

$$0 =_\varepsilon C'(R_0, \ldots, R_n; \varepsilon) + \sum_{i=1}^{d_{C'}} A^i \cdot \left(\frac{\partial_{x_n^i}(C')(R_0, \ldots, R_n)}{i!} \right) \bmod \langle \mathbf{z} \rangle^{j+1}$$

Moreover, since $A = \Delta_n(P_{n+j})(\mathbf{z}, \mathbf{a})$ is a homogeneous polynomial of degree $j \geq 1$, we have $A^2 = 0 \bmod \langle \mathbf{z} \rangle^{j+1}$. Therefore,

$$\begin{aligned} 0 &=_\varepsilon C'(R_0, \ldots, R_n; \varepsilon) + \sum_i A^i \cdot \left(\frac{\partial_{x_n^i}(C')(R_0, \ldots, R_n)}{i!} \right) \bmod \langle \mathbf{z} \rangle^{j+1} \\ &= C'(R_0, \ldots, R_n; \varepsilon) + A \cdot \left(\partial_{x_n}(C')(R_0, \ldots, R_n) \right) \bmod \langle \mathbf{z} \rangle^{j+1} \\ &= C'(R_0, \ldots, R_n; \varepsilon) + A \cdot \alpha \bmod \langle \mathbf{z} \rangle^{j+1} \end{aligned}$$

where $\alpha = \partial_{x_n}(C')(R_0, \ldots, R_n; \varepsilon)(\mathbf{0})$, the constant term of $\partial_{x_n}(C')(R_0, \ldots, R_n; \varepsilon)(\mathbf{z})$.

$$\begin{aligned} \alpha = \partial_{x_n}(C')(R_0, \ldots, R_n; \varepsilon)(\mathbf{0}) &= \partial_{x_n}(C')(\Delta_0(P_{\leq n+j-1})(\mathbf{0}, \mathbf{a}), \ldots, \Delta_n(P_{\leq n+j-1})(\mathbf{0}, \mathbf{a}); \varepsilon) \\ &= \partial_{x_n}(C')(\Delta_0(P_{\leq n+j-1})(\mathbf{z}, \mathbf{a}), \ldots, \Delta_n(P_{\leq n+j-1})(\mathbf{z}, \mathbf{a}); \varepsilon)(\mathbf{0}) \\ &= \partial_{x_n}(C')(\Delta_0(P)(\mathbf{z}, \mathbf{a}), \ldots, \Delta_n(P)(\mathbf{z}, \mathbf{a}); \varepsilon)(\mathbf{0}) \\ &= \left(\partial_{x_n}(C') \circ \mathsf{Gen}(P, \mathbf{a}) \right)(\mathbf{0}) \\ &= \Psi(\mathbf{a}; \varepsilon) \neq_\varepsilon 0 \end{aligned}$$

Combining this with the previous equation, we get

$$\begin{aligned} 0 &=_\varepsilon C'(R_0, \ldots, R_n) + A \cdot \Psi(\mathbf{a}) \bmod \langle \mathbf{z} \rangle^{j+1} \\ \implies A = \Delta_n(P_{n+j})(\mathbf{z}, \mathbf{a}) &=_\varepsilon \left(\frac{-1}{\Psi(\mathbf{a})} \right) \cdot C'(R_0, \ldots, R_n) \bmod \langle \mathbf{z} \rangle^{j+1}. \end{aligned}$$

□(Lemma 3.2)

3.2 Application to bootstrapping phenomenon for hitting sets

We now use Theorem 1.6 to prove the following theorem about bootstrapping hitting sets for algebraic circuits. The main differences of this result from the earlier results of this flavor is that the bootstrapping here is done in one step, and the final running time is truly polynomially bounded, whereas the earlier proofs had a iterative argument for stretching the number of variables, and the final running time was of the form $s^{2^{2^{O(\log^* n)}}}$. Other crucial difference is that in the result below, we need to work with the border of polynomials with small circuits.

Theorem 1.7 (Bootstrapping in one shot)**.** *Assume that the underlying field $\mathbb{F}$ has characteristic zero. Let $\delta > 0$ be any constant and $k \in \mathbb{N}$ be a large enough constant. Suppose that, for all large enough s, there is an $s^{O(k)}$-explicit hitting set of size $s^{k-\delta}$ for all degree s polynomials which are in the border of size s algebraic circuits over k variables. Then, there is an $s^{O(k^3)}$-explicit hitting set of size $s^{O(k^3)}$ for all of degree s polynomials which are in the border of size s algebraic circuits over s variables.*

Proof. Let $s' = s^{40k^2/\delta}$. Let H be the hitting set guaranteed by the hypothesis of the theorem for k-variate polynomials that are ε-computed by size s' and degree s' circuits. Since H is a set of size at most $s'^{k-\delta}$, there is a k-variate polynomial $P(\mathbf{z})$ of individual degree at most $s'^{(k-\delta)/k}$ that vanishes on H. By Theorem 2.2, the polynomial $P(\mathbf{z})$ cannot be ε-computed by circuits of size s'.

Now suppose $0 \neq_\varepsilon C(\mathbf{x}; \varepsilon)$ is an s-variate, degree s circuit of size at most s. Then, by Theorem 1.6, if $C \circ \mathsf{Gen}_P(\mathbf{z}, \mathbf{y}) =_\varepsilon 0$, then $P(\mathbf{z})$ can be ε-computed by circuits of size at most

$$s \cdot s \cdot k s'^{(k-\delta)/k} \cdot s^{10k} \leq s' \cdot \left(\frac{s^{20k}}{s'^{\delta/k}}\right) = s' \cdot \left(\frac{s^{20k}}{s^{40k}}\right) < s'$$

which contradicts the hardness of P. Hence, it must be the case that

$$Q = \lim_{\varepsilon \to 0} (C \circ \mathsf{Gen}_P(\mathbf{z}, \mathbf{y})) \neq 0.$$

Note that Q is a non-zero k-variate polynomial of degree at most $s \cdot k \cdot s' \leq s^{50k^2/\delta}$. Thus, by composing the generator $\mathsf{Gen}_P(\mathbf{z}, \mathbf{y})$ with the trivial hitting set from Lemma 2.1, we have a hitting set of size at most $s^{50k^3/\delta}$ for C. $\square$

4 Open Problems

We end with some open problems.

- The construction of the HSG in this paper needs the characteristic of the field to be large enough or zero. Constructing a HSG with similar properties (seed length, stretch, running time, degree) over fields of small positive characteristic would be quite interesting.

- The role of approximative or border computation in the analysis of the HSG here is quite intriguing. Thus, as of now, Question 1.3 and Question 1.4 as stated continue to remain open. It would be interesting to construct a HSG with properties similar to the one in this paper which does not go via the border.

- In the current statement of Theorem 1.6, the hardness required for P for the HSG to fool circuits of size s, depends on the degree of this circuit. We suspect that this dependence on the degree can be avoided, and in particular, this HSG should fool all circuits of small size regardless of their degree.

- Lastly, it would be interesting to understand if this new HSG and the ideas in its analysis have any other applications.

Acknowledgements

We thank Marco Carmosino, Chi-Ning Chou, Nutan Limaye, Rahul Santhanam, Srikanth Srinivasan and Anamay Tengse for insightful conversations at various stages of this work. We also thank Madhu Sudan for many insightful discussions and much encouragement; in particular for patiently sitting through a presentation of a preliminary version of the proof of Lemma 3.2.

Mrinal is also thankful to Swastik Kopparty for many helpful discussions on Nisan-Wigderson generator and list decoding algorithms for multiplicity codes while he was a PhD student at Rutgers. Swastik's interest in a purely algebraic solution for algebraic hardness randomness tradeoffs had a non-trivial role in motivating this work.

References

[Agr05] Manindra Agrawal. Proving Lower Bounds Via Pseudo-random Generators. In *Proceedings of the 25th International Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2005)*, pages 92–105, 2005.

[AGS18] Manindra Agrawal, Sumanta Ghosh, and Nitin Saxena. Bootstrapping variables in algebraic circuits. In *Proceedings of the 50th Annual ACM Symposium on Theory of Computing (STOC 2018)*, pages 1166–1179, 2018. `eccc:TR18-035`.

[BIZ18] Karl Bringmann, Christian Ikenmeyer, and Jeroen Zuiddam. On Algebraic Branching Programs of Small Width. *J. ACM*, 65(5):32:1–32:29, 2018.

[DL78] Richard A. DeMillo and Richard J. Lipton. A Probabilistic Remark on Algebraic Program Testing. *Information Processing Letters*, 7(4):193–195, 1978.

[FSV18] Michael A. Forbes, Amir Shpilka, and Ben Lee Volk. Succinct Hitting Sets and Barriers to Proving Lower Bounds for Algebraic Circuits. *Theory of Computing*, 14(1):1–45, 2018. Preliminary version in the *49th Annual ACM Symposium on Theory of Computing (STOC 2017)*.

[GKSS17] Joshua A. Grochow, Mrinal Kumar, Michael E. Saks, and Shubhangi Saraf. Towards an algebraic natural proofs barrier via polynomial identity testing. *CoRR*, abs/1701.01717, 2017. Pre-print available at `arXiv:1701.01717`.

[HS80] Joos Heintz and Claus-Peter Schnorr. Testing Polynomials which Are Easy to Compute (Extended Abstract). In *Proceedings of the 12th Annual ACM Symposium on Theory of Computing (STOC 1980)*, pages 262–272, 1980.

[Kal89] Erich Kaltofen. Factorization of Polynomials Given by Straight-Line Programs. In *Randomness and Computation*, pages 375–412. JAI Press, 1989.

[KI04] Valentine Kabanets and Russell Impagliazzo. Derandomizing Polynomial Identity Tests Means Proving Circuit Lower Bounds. *Computational Complexity*, 13(1-2):1–46, 2004. Preliminary version in the *35th Annual ACM Symposium on Theory of Computing (STOC 2003)*.

[Kop15] Swastik Kopparty. List-Decoding Multiplicity Codes. *Theory of Computing*, 11(5):149–182, 2015.

[KST19] Mrinal Kumar, Ramprasad Saptharishi, and Anamay Tengse. Near-optimal Bootstrapping of Hitting Sets for Algebraic Circuits. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 639–646, 2019.

[Kum18] Mrinal Kumar. On top fan-in vs formal degree for depth-3 arithmetic circuits. Unpublished manuscript. `https://mrinalkr.bitbucket.io/papers/topfanin-formaldeg.pdf`, 2018.

[NW94] Noam Nisan and Avi Wigderson. Hardness vs Randomness. *Journal of Computer and System Sciences*, 49(2):149–167, 1994. Available on `citeseer:10.1.1.83.8416`.

[Ore22] Øystein Ore. Über höhere Kongruenzen. *Norsk Mat. Forenings Skrifter*, 1(7):15, 1922.

[Sch80] Jacob T. Schwartz. Fast Probabilistic Algorithms for Verification of Polynomial Identities. *Journal of the ACM*, 27(4):701–717, 1980.

[SU05] Ronen Shaltiel and Christopher Umans. Simple Extractors for All Min-entropies and a New Pseudorandom Generator. *J. ACM*, 52(2):172–216, March 2005.

[Uma03] Christopher Umans. Pseudo-random generators for all hardnesses. *Journal of Computer and System Sciences*, 67(2):419–440, 2003.

[Zip79] Richard Zippel. Probabilistic algorithms for sparse polynomials. In *Symbolic and Algebraic Computation, EUROSAM '79, An International Symposiumon Symbolic and Algebraic Computation*, volume 72 of *Lecture Notes in Computer Science*, pages 216–226. Springer, 1979.

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il