Electronic Colloquium on Computational Complexity, Revision 1 of Report No. 60 (2017)

Limits on Low-Degree Pseudorandom Generators (Or: Sum-of-Squares Meets Program Obfuscation)

Boaz Barak * Zvika Brakerski † Ilan Komargodski ‡ Pravesh K. Kothari §

Abstract

An m output pseudorandom generator $\mathcal{G}\colon (\{\pm 1\}^b)^n \to \{\pm 1\}^m$ that takes input n blocks of b bits each is said to be ℓ-block local if every output is a function of at most ℓ blocks. We show that such ℓ-block local pseudorandom generators can have output length at most $\tilde{O}(2^{\ell b} n^{\lceil \ell/2 \rceil})$, by presenting a polynomial time algorithm that distinguishes inputs of the form $\mathcal{G}(x)$ from inputs where each coordinate is sampled from the uniform distribution on m bits.

As a corollary, we refute some conjectures recently made in the context of constructing provably secure indistinguishability obfuscation (iO). This includes refuting the assumptions underlying Lin and Tessaro's [LT17] recently proposed candidate iO from bilinear maps. Specifically, they assumed the existence of a secure pseudorandom generator $\mathcal{G}\colon \{\pm 1\}^{nb} \to \{\pm 1\}^{2^{cb} n}$ as above for large enough $c > 3$ and $\ell = 2$. (Following this work, and an independent work of Lombardi and Vaikuntanthan [LV17b], Lin and Tessaro retracted the bilinear maps based candidate from their manuscript.)

Our results actually hold for the much wider class of low-degree, non-binary valued pseudorandom generators: if every output of $\mathcal{G}\colon \{\pm 1\}^n \to \mathbb{R}^m$ ($\mathbb{R}$ = reals) is a polynomial (over $\mathbb{R}$) of degree at most d with at most s monomials and $m \geqslant \tilde{\Omega}(s n^{\lceil d/2 \rceil})$, then there is a polynomial time algorithm for distinguishing the output $\mathcal{G}(x)$ from z where each coordinate z_i is sampled independently from the marginal distribution on $\mathcal{G}_i$. Furthermore, our results continue to hold under arbitrary *pre-processing* of the seed. This implies that any such map $\mathcal{G}$, with arbitrary seed pre-processing, cannot be a pseudorandom generator in the mild sense of fooling a product distribution on the output space. This allows us to rule out various natural modifications to the notion of generators suggested in other works that still allow obtaining indistinguishability obfuscation from bilinear maps.

Our algorithms are based on the Sum of Squares (SoS) paradigm, and in most cases can even be defined more simply using a canonical semidefinite program. We complement our algorithm by presenting a class of candidate generators with block-wise locality 3 and constant block size, that resists both Gaussian elimination and sum of squares (SOS) algorithms whenever $m = n^{1.5-\varepsilon}$. This class is extremely easy to describe: Let $\mathbb{G}$ be any simple non-abelian group with the group operation "$*$", and interpret the blocks of x as elements in $\mathbb{G}$. The description of the pseudorandom generator is a sequence of m triples of indices (i, j, k) chosen at random and each output of the generator is of the form $x_i * x_j * x_k$.

*Harvard University, b@boazbarak.org. Supported by NSF awards CCF 1565264 and CNS 1618026, and the Simons Foundation. Work done while the author visited Weizmann Institute during Spring 2017.

†Weizmann Institute of Science, zvika.brakerski@weizmann.ac.il. Supported by the Israel Science Foundation (Grant No. 468/14) and Binational Science Foundation (Grants No. 2016726, 2014276).

‡Cornell Tech, NYC. komargodski@cornell.edu. Supported in part by Elaine Shi's Packard Foundation Fellowship. Work done while the author was a Ph.D student at the Weizmann Institute of Science, supported in part by a grant from the Israel Science Foundation and by a Levzion Fellowship.

§Princeton University and IAS kothari@cs.princeton.edu. Work done while the author visited Weizmann Institute in March 2017.

ISSN 1433-8092

Contents

1 Introduction

Understanding how "simple" a pseudorandom generator can be has been of great interest in cryptography and computational complexity. In particular, researchers have studied the question of whether there exist pseudorandom generators with *constant input locality*, in the sense that every output bit only depends on a constant number of the input bits. Applebaum, Ishai and Kushilevitz [AIK06] showed that, assuming the existence of one-way functions computable by log-depth circuits, there is such a generator mapping n bits to $n + n^{\varepsilon}$ bits for a small constant $\varepsilon > 0$. Goldreich [Gol00] gave a candidate pseudorandom generator of constant locality that could potentially have even *polynomially large* stretch (e.g. map n bits to n^s bits for some $s > 1$).[1] The possibility of such "ultra simple" high-stretch pseudorandom generators has attracted significant attention recently with applications including:

- Public key cryptography from "combinatorial" assumptions [ABW10].
- Highly efficient multiparty computation [IKO+11].
- Reducing the assumptions needed for constructing *indistinguishability obfuscators* (iO) [AJS15, Lin16a, LV16, Lin16b, AS16, LT17].

The last application is perhaps the most exciting, as it represents the most promising pathway for basing this important cryptographic primitive on more standard assumptions. Furthermore, this application provides motivation for considering qualitatively different notions of "simplicity" of a generator. For example, it is possible to relax the condition of having small input locality to that of just having small algebraic *degree* (over the rationals), as well as allow other features such as preprocessing of the input and admitting non-Boolean outputs.

At the same time, the application to obfuscation emphasizes a fine-grained understanding of the quantitative relationship between the "simplicity" of a generator (such as its locality, or algebraic degree) and its *stretch* (i.e., ratio of output and input lengths). For example, works of Lin and Ananth and Sahai [Lin16b, AS16] show that a generator mapping n bits to $n^{1+\varepsilon}$ bits with locality 2 implies an obfuscation candidate based on standard cryptographic assumptions – a highly desired goal, but it is known that it is impossible to achieve super-linear stretch with locality four (let alone two) generator [MST06].

Very recently, Lin and Tessaro [LT17] proposed bypassing this limitation by considering a relaxation of locality to a notion they referred to as *block locality*. They also proposed a candidate generator with the required properties. If such secure PRGs exist, this would imply obfuscators whose security is based on standard cryptographic assumptions, a highly desirable goal. Ananth et al. [ABKS17] observed that the conditions can be relaxed further to allow generators without a block structure, and even allow non-Boolean outputs, but their method requires (among other restrictions) that each output is computed by a sparse polynomial of small degree.

In this paper we give strong limitations on this approach, in particular giving negative answers to some of the questions raised in prior works. While a priori, questions of algebraic flavor, such as the difference between the power of bilinear vs trilinear maps, and those of combinatorial essence such as the difficulty of refuting random constraint satisfaction instances might seem unrelated,

[1] While Goldreich originally only conjectured that his function is a one-way function, followup work has considered the conjecture that it is a pseudorandom generator, and also linked the two questions (see e.g., [App13, AR16]; see also Applebaum's survey [App16]).

it turns out that techniques useful in the study of CSP refutation yield a barrier that, somewhat surprisingly, seems to exactly correspond to what is needed to bypass the "trilinear map barrier" for obfuscation constructions.

We complement our negative results with a simple construction of a candidate degree *three* pseudorandom generator which resists known attacks (Gaussian elimination and sum-of-squares algorithms) even for output length $n^{1+\Omega(1)}$.

1.1 Our results

To state our results, let us define the notion of the *image refutation problem* for a map $\mathcal{G}$ that takes n inputs into m outputs (e.g., a purported pseudorandom generator). Looking ahead, we will allow maps to have non-Boolean outputs.[2] Informally, the image refutation problem asks for a efficiently computable certificate for a random string *not* being in the image of a purported generator $\mathcal{G}$.

Definition 1.1 (Refutation problem)**.** Let $\mathcal{G}\colon \{\pm 1\}^n \to \mathbb{R}^m$ and Z be a distribution over $\mathbb{R}^m$. An algorithm A is said to solve the *$\mathcal{G}$-image refutation problem w.r.t Z* if on input $z \in \mathbb{R}^m$, A outputs either "`refuted`" or "?" and satisfies:

- If $z = \mathcal{G}(x)$ for some $x \in \{\pm 1\}^n$ then $A(z) =$ "?".
- $\mathbb{P}_{z\sim Z}[A(z) =$ "`refuted`"$] \geqslant 0.5$

Note that in particular if Z is the uniform distribution over $\{0,1\}^m$, then the existence of an efficient algorithm that solves the $\mathcal{G}$ image refutation problem with respect to Z means that $\mathcal{G}$ is not a pseudorandom generator - in fact, an image refutation algorithm, with probability at least $1/2$, shows that a random string from $\{\pm 1\}^m$ is not in the image of $\mathcal{G}$.

Remark 1.2 (Refutation vs Distinguishing). It is instructive to contrast the algorithmic tasks of image refutation with the easier task of distinguishing the output of a pseudorandom generator from a uniformly random string. In the latter case, we are typically concerned with distinguishing the output distribution of a generator $\mathcal{G}\colon \{\pm 1\}^n \to \{\pm 1\}^m$ when the input is chosen according to the uniform distribution on $\{\pm 1\}^m$. It's easy to see that a refutation algorithm immediately yields a distinguisher. In general, refutation, however can be more powerful. For example, a refutation algorithm can distinguish between the uniform distribution on $\{\pm 1\}^m$ from the output distribution of the generator even under *arbitrary* distributions on the seed. Thus, an image refutation algorithm rules out not only the natural PRG construction but also natural modifications that involve using some non-trivial pre-processing on the seed before inputting it into the generator, thus modifying the input distribution. Such modifications were in fact suggested for candidate constructions of iO from bilinear maps in the concurrent work of [LV17b]. While a distinguisher for the original PRG may fail after this modification, a refutation algorithm continues to work. As we discuss later, this is one of the key differences in our approach from that of [LV17b].

Our first result is a limitation on generators with "block locality" two:

Theorem 1.3 (Limitations of two block local generators)**.** *For every n, b, let $\mathcal{G}\colon \{\pm 1\}^{nb} \to \{\pm 1\}^m$ be such that, if we partition the input into n blocks of size b, then every output of G depends only on variables*

[2] Allowing non-Boolean output can make a significant difference. For example, [LV17a, Theorem 6.1] show that every degree two Boolean-valued function on $\{\pm 1\}^n$ depends on at most four variables, which in particular means that it cannot be used as the basis for a pseudorandom generator with super-linear output length. It also allows us to consider polynomials that only take the values in $\{\pm 1\}$ on a subset of their inputs.

inside two blocks. Then, there is an absolute constant K such that if $m > K \cdot 2^{2b} n \log^2 n$, then there is an efficient algorithm for the $\mathcal{G}$-image-refutation problem w.r.t. the uniform distribution over $\{\pm 1\}^m$.

Theorem 1.3 yields an attack on the aforementioned candidate pseudorandom generator proposed by Lin and Tessaro [LT17] towards basing indistinguishability obfuscator on bilinear maps, as well as any other candidate of block-locality 2 compatible with their construction.

A special case that has been of considerable interest in literature is one where all outputs of the PRG are computed by the same two-block-local predicate $P\colon \{\pm 1\}^b \to \{\pm 1\}^b \to \{\pm 1\}$. For this case, we give an image refutation algorithm that works whenever the stretch $m = \tilde{\Omega}(n2^b)$. [3]

Theorem 1.4 (Limitations of two block local generators with a single predicate, Theorem 5.3)**.** *For every n, b, let $\mathcal{G}\colon \{\pm 1\}^{nb} \to \{\pm 1\}^m$ be such that, if we partition the input into n blocks of size b, then every output of $\mathcal{G}$ is the same predicate P applied to two b-bit blocks. Then, there is an absolute constant K such that if $m > K \cdot 2^b n \log^2 n$, then there is an efficient algorithm for the $\mathcal{G}$-image-refutation problem w.r.t. the uniform distribution over $\{\pm 1\}^m$.*

Yet another special case of interest is where the candidate generator obtained is chosen at random: that is, the m pairs of blocks used to compute the output are chosen at random and, further, each predicate computing an output is chosen randomly and independently conditioned on being balanced. For this case, we show (in Theorem 5.16, Section 5.3) that we can again improve our bound on the output length from $\tilde{O}(2^{2b} n)$ to $\tilde{O}(2^b n)$:

Our next result applies to any degree d map, and even allows maps with non-Boolean output. For the refutation problem to make sense, the probability distribution Z must be non-degenerate or have large entropy, as otherwise it may well be the case that $z \sim Z$ is in the image of $\mathcal{G}$ with high probability. For real-valued distributions, a reasonable notion of non-degeneracy is that the distribution does not fall inside any small interval with high probability. Specifically, if we consider *normalized* product distributions (where $\mathbb{E} Z_i = 0$ and $\mathbb{E} Z_i^2 = 1$ for every i and the Z_i are independent), then we say that Z is *c-spread* (see Definition 4.1) if it is a product distribution and $\mathbb{P}[Z_i \notin I] \geqslant 0.1$ for every interval $I \subseteq \mathbb{R}$ of length at most $1/c$ (where we can think of c as a large constant or even a poly-logarithmic or small polynomial factor).

If Z is supposed to be indistinguishable from $\mathcal{G}(U)$, where U is the uniform distribution over $\{\pm 1\}^n$, then these two distributions should agree on the marginals and in particular at least on their first and second moments. Hence, we can assume that the map $\mathcal{G}$ has the same normalization as Z, meaning that $\mathbb{E}\, \mathcal{G}(U)_i = 0$ and $\mathbb{E}\, \mathcal{G}(U)_i^2 = 1$.[4] Our result for general low degree generators is the following:

Theorem 1.5 (Limitations on degree d generators)**.** *Suppose that $\mathcal{G}\colon \{\pm 1\}^n \to \mathbb{R}^m$ is such that for every $i \in [m]$ the map $x \mapsto \mathcal{G}(x)_i$ is a normalized polynomial of degree at most d with at most s monomials. Let Z be a c-spread product distribution over $\mathbb{R}^m$. Then, there is some absolute constant K such that if $m \geqslant Kc^2 s n^{\lceil d/2 \rceil} \log^2 n$, then there is an efficient algorithm for the $\mathcal{G}$-image-refutation problem w.r.t. Z.*

We believe the dependence on the degree d can be improved in the odd case from $\lceil d/2 \rceil$ to $d/2$. Resolving this is related to some problems raised in the CSP refutation literature (e.g., see [Wit17, Questions 5.2.3,5.2.7,5.2.8]).

[3] Unlike the other results in this paper, Theorem 1.4 builds upon the concurrent work [LV17b]. See Section 1.3 for a detailed comparison between this work and [LV17b].

[4] We say that $\mathcal{G}$ is *normalized* if it satisfies these conditions. Clearly, any map can be normalized by appropriate shifting and scaling.

While for arbitrary polynomials we do not know how to remove the restriction on sparsity (i.e., number of non-zero monomials s), we show in Section 4 that we can significantly relax it in several settings. Moreover, the applications to obfuscation require generators that are both low degree and sparse; see Section 2. Nevertheless, we view eliminating the dependence on the sparsity as the main open question left by this work. We conjecture that this can be done, at least in the pseudorandom generator setting, as paradoxically, it seems that the only case where our current algorithm fails is when the pseudorandom generator exhibits some "non-random" behavior. Improving this is related to obtaining better upper bound on the stretch of block-local generators.

Up to the dependence on sparsity, Theorem 1.5 answers negatively a question of Lombardi and Vaikuntanathan [LV17a, Question 7.2], who asked whether it is possible to have a degree d pseudorandom generator with stretch $n^{\lceil \frac{3}{4}d \rceil + \varepsilon}$. It was already known by the work of Mossel et al. [MST06] that such output length cannot be achieved by *d-local* generators; our work shows that, at least for $n^{o(1)}$-sparse polynomials, relaxing locality to the notion of algebraic degree does not help achieve a better dependency .

All of our results are based on the same algorithm: the *sum of squares* (SOS) semidefinite program ([Sho87, Par00, Las01]; see the lecture notes [BS17]). This is not surprising as for refuting CSPs, semidefinite programs in general and the sum-of-squares semi-definite programming hierarchy in particular are the strongest known general tools [RRS16, KMOW17]. This suggests that for future candidate generators, it will be useful to prove resilience at least with respect to this algorithm. Fortunately, there is now a growing body of techniques to prove such lower bounds.

Here, we establish that the sum-of-squares algorithm cannot be used to give an attack on PRGs with stretch $O(n2^b)$. Note that the sum of squares algorithm captures all the techniques in literature for efficiently refuting random CSPs including the algorithms in this paper and the work of [LV17b]. Our lower bound on the sum of squares algorithm below shows that using such techniques, one cannot hope to attack two-block-local PRGs with stretch at most $O(n2^b)$ - for the case of identical predicates computing all outputs of the generator, this, in particular, establishes the optimality of our analyis of any technique captured by the sum of squares framework.

Concretely, in Section 6, we show that there is a natural sum-of-squares resistant construction with a stretch of $\tilde{\Theta}(n2^b)$.

Theorem 1.6 (See Theorem 6.1 for a formal version)**.** *For any $b \geqslant 10 \log\log(n)$, there is a construction of a two-block-local PRG $\mathcal{G}\colon (\{\pm 1\}^b)^n \to \{\pm 1\}^m$ for $m = \Omega(n2^b)$ such that degree-$\Theta(n/2^{4b})$ sum of squares algorithm cannot solve the refutation problem for $\mathcal{G}$.*

For example, for $b < \varepsilon/4 \log(n)$, the above results rules out an attack on $\Omega(n2^b)$-stretch PRGs using SoS algorithm that runs in time $\sim 2^{n^{1-\varepsilon}}$.

While our results give strong barriers for degree *two* pseudorandom generators, they do not rule out a degree *three* pseudorandom generator with output length $n^{1+\Omega(1)}$. Indeed, we show a very simple candidate generator that might satisfy this property. This is the generator $\mathcal{G}$ mapping $\mathbb{G}^n$ to $\mathbb{G}^m$ where $\mathbb{G}$ is some finite *non-abelian* simple group (e.g., the size 60 group A_5), where for every $\ell \in [m]$, the ℓ^{th} output of $\mathcal{G}(x)$ is obtained as

$$\mathcal{G}(x)_\ell = x_i * x_j * x_k$$

for randomly chosen indices i, j, k and $*$ is the group operation. This generator has block locality three with constant size blocks and also (using the standard representation of group elements as matrices) has algebraic degree three as well. Yet, it is a hard instance for the SOS algorithm which

encapsulates all the techniques used in this paper. While more study of this candidate's security is surely needed, there are results suggesting that it resists algebraic attacks such as Gaussian elimination [GR02]. See Section 7 for details.

1.2 Prior works

Most prior works on limitations of "simple" pseudorandom generators focused on providing upper bounds on the output length in terms of the *locality*. Cryan and Miltersen [CM01] observe that there is no PRG with locality 2 and proved that there is no PRG with locality 3 achieving super linear stretch (i.e., having input length n and output length $n + \omega(n)$ bits). Mossel, Shpilka, and Trevisan [MST06] extended this result to locality 4 PRGs and constructed (non-cryptographic) small-biased locality 5 generators with linear stretch and exponentially-small bias. They also showed that a k local generator cannot have output length better than $O(2^k n^{\lceil k/2 \rceil})$. Applebaum, Ishai, and Kushilevits [AIK06] showed that, under standard cryptographic assumptions, there are locality 4 PRGs with sublinear-stretch. Applebaum and Raykov [App13, AR16] related the pseudorandomness and one-wayness of Goldreich's proposed one-way function [Gol00] in some regime of parameters.

Apart from our focus on *degree* in the place of locality, another feature that distinguishes our work from much of the prior works on pseudorandom generators is the focus on the *refutation* problem (certifying that a random string is *not* in the image of the generator) as opposed to the *search* problem (given the output of a uniformly random seed, recover the seed). This is important for us since we do not want to make the typical assumption that the input (i.e., seed) to the pseudorandom generator is uniformly distributed, as to allow the possibility of preprocessing for it.

The refutation problem was extensively studied in the context of random *constraint satisfaction problems* (CSPs). The refutation problem for a k-local generator with n inputs and m outputs corresponds to refuting a CSP with n variables and m constraints. Thus, the study of limitations for local generators is tightly connected to the study of refutation algorithm for CSPs. Most well studied in this setting is the problem of refuting random CSPs - given a random CSP instance with a predicate P, certify that it is far from satisfiable with high probability. There is a large body of works on the study of refuting *random* and *semirandom* CSPs, starting with the work of Feige [Fei02].[5]

In particular, we now know tight relations between the *arity* (or locality) of the predicates and the number of constraints required to refute random instances [AOW15a, RRS16, KMOW17] using the sum-of-squares semidefinite programming hierarchy - the algorithm of choice for the problem.

Most relevant to the current paper are works from this literature that deal with predicates that have large arity but have small degree d (or the related notion of not supporting $(d+1)$-wise independent distribution). Allen, O'Donnell, and Witmer [AOW15a] showed that random instances of such predicates can be refuted when the number of constraints m is larger than $\tilde{O}(k^d n^{d/2})$. In his thesis proposal, Witmer [Wit17] sketched how to generalize this to the *semirandom* setting, though only for the case of *even* degree d. This is related to the questions considered in this work for higher degree, though our model is somewhat more general, considering not just CSPs but arbitrary low-degree maps.

The notion of ℓ *block locality* is equivalent to the notion of CSPs of arity ℓ over a *large alphabet* (specifically, exponential in the block size). Though much of the CSP refutation and approximation

[5] In a *random* CSP the graph of dependence between variables and constraints is random, and we also typically consider adding a random pattern of negations or shifts to either the inputs or the outputs of the predicates. In *semirandom* instances [Fei07, FO07], the graph is arbitrary and only this pattern of negations or shifts is random.

literature deals with CSPs over a binary alphabet, there have been works dealing with larger alphabet (see e.g., [AOW15a]). The work of [BRS11] gives an SOS based algorithm for 2-local CSPs over large alphabet (or equivalently, 2 block-local CSPs) as long as the underlying constraint graph is a sufficiently good expander. However, their algorithm (at least their analysis) has an *exponential* dependence in the running time on the alphabet size which is unsuitable for our applications.

The main technical difference between our work and prior results in the CSP literature, is that since for CSPs we often think as the arity as constant, these works often had poor dependence on this parameter, whereas we want to handle the case that it can be as large as n^{ε} or in some cases even unrestricted. Another difference is that in the cryptographic setting, we wish to allow the designer of a pseudorandom generator significant freedom, and this motivates studying more challenging semirandom models than those typically used in prior works. We discuss these technical issues in more depth in Section 3.

The algorithms in almost all the refutation works in the CSP literature can be encapsulated by the *sum of squares* semidefinite programming hierarchy. Some lower bounds for this hierarchy, showing tightness of these analysis, were given in [BCK15, OW14, KMOW17]. For the alphabet-size sensitive setting of block-local PRGs, we give a lower bound in Section 6.

1.3 Comparison with [LV17b]

In a concurrent and independent work, Lombardi and Vaikuntanathan [LV17b] also analyzed the possibility of a secure block-wise local PRG motivated by the work of Lin and Tessaro [LT17]. They show that there exists an efficient polynomial-time distinguisher with the following property: for any $m \geqslant \tilde{\Omega}(n2^b)$ and any predicate $P\colon \{\pm 1\}^b \times \{\pm 1\}^b \to \{\pm 1\}$ in two blocks of size b, there's an efficient distinguishing algorithm for the following two distributions over $\{\pm 1\}^m$: 1) the uniform distribution on $\{\pm 1\}^m$ and 2) the output distribution of Goldreich's PRG $G_H\colon (\{\pm 1\}^b)^n \to \{0,1\}^m$ instantiated with a random graph H and the single predicate P computing all m outputs when given a uniformly random nb bit string as input. [6]

We point out the major differences between our results on block-local PRGs and that of [LV17b] here.

1. *Distinguishing vs Refutation:* As discussed in Remark 1.2, our approach yields the stronger refutation guarantees while that of [LV17b] yields a distinguisher. This allows us to show that reinforcing the block-local (or low-degree, more generally) PRGs by allowing arbitrary input preprocessing cannot lead to a larger stretch. This is important, as preprocessing is OK to do in the context of the applications for obfuscation, and in fact this was one of the avenues suggested for bypassing these general type of negative results.

2. *Single Predicates vs Multiple Predicates:* The work of [LV17b] only applies to the PRGs where each output is computed using the *same* predicate. Our approach shows that block-local (or low-degree) PRGs cannot achieve large enough stretch even if each output is computed using a different predicate - a priori, one could hope that using different predicates for different outputs could add significantly to the stretch of the PRG. This bottleneck is in fact inherent in the technical approach of [LV17b]. In particular, our approach allows us to analyze the natural candidate for 2-block-local generator obtained by applying independently chosen

[6] From private communication with [LV17b], we learned that in an updated version of their work, they use a refutation algorithm from our work to extend their distinguisher to the case when the graph H is arbitrarily chosen.

multiple random predicates to randomly chosen pairs of input blocks and yields an $\tilde{O}(2^b n)$ upper bound on their stretch, see Section 5.3.

3. *Random Graph vs Arbitrary Graphs:* The work of [LV17b] only handles block-local PRGs when the underlying graph G defining the generator is chosen at random. This was because [LV17b] relied on CSP refutation results that work under the assumption of the instance being random.

4. *Special Case of Single Predicate Block-Local PRGs:* For the PRGs with all outputs computed by a single predicate, [LV17b] show a distinguisher that works whenever the stretch of the PRG is $\Omega(n2^b)$. For this case, we show that our algorithm in fact guarantees image *refutation* at the same stretch requirement. (A previous version of our work didn't include this result on PRGs with single predicate.) Our refutation algorithm (Theorem 1.4) is in fact inspired by the application of the Chor-Goldreich Lemma in the work of [LV17b].

We note that the three first differences: image refutation as opposed to distinguishing, allowing different predicates as opposed to a single predicate, and using arbitrary graphs as opposed to random graphs, exactly correspond to the open questions raised by [LV17b].[7] Thus, our results block all the approaches that [LV17b] identified as potential strategies for repairing the iO candidate. This suggests that, rather than a "patchable problem", there is perhaps a fundamental barrier to this approach of obtaining iO from bilinear maps.

1.4 Paper organization

Section 2 explains the connection between simple generators and the construction of indistinguishability obfuscator. This explanation allows us to draw the conclusion that our algorithm renders recently proposed methods ineffective for constructing obfuscation from standard cryptographic assumptions. For those interested in additional details, Appendix B) contains more information about constructing obfuscators and in particular on the new result of [LT17]. In Section 3, we provide a high level overview of our algorithmic techniques. Section 4 contains our main algorithm and analysis, and in particular proves Theorem 1.5. We use standard tools from the SDP/SOS literature that can be found in Appendix A. In Section 5 we focus our attention on pseudorandom generators with small block-locality and show tighter results than those achieved by our general analysis, in particular we prove Theorem 1.3 as well as an even tighter result for generators with single predicates (Theorem 5.3) and random two-block-local PRGs (Theorem 5.16). In Section 6, we show that sum-of-squares algorithm cannot be used to prove sharper upper bounds on the stretch than $\sim n2^b$. Finally, in Section 7 we present our class of candidate block-local generators.

2 Relating simple generators and program obfuscators

A program obfuscator [Had00, BGI+01] is a compiler that given a program (say represented as a Boolean circuit) transforms it into another "scrambled" program which is functionally equivalent but its implementation details are "hidden", making it hard to reverse-engineer. The study of *indistinguishability obfuscation* (iO) stands at the forefront of cryptographic research in recent years due to two main developments. Firstly, Garg et al. [GGH+13b] suggested that this notion might be achievable given sufficiently strong *cryptographic multilinear maps*, for which a candidate construction

[7] See Section 5 on page 12 of https://eprint.iacr.org/2017/301/20170409:183008.

was given by [GGH13a]. Secondly, it was shown by Sahai and Waters [SW14] and numerous follow-up works that iO is extremely useful for constructing a wide variety of cryptographic objects, many of which are unknown to exist under any other assumption.

A fundamental question in the construction of iO from multilinear maps is the *level of multilinearity*. Without going into details, this essentially corresponds to the highest degree of polynomials that can be evaluated by this object. Whereas multilinear maps of level 2, a.k.a *bilinear maps*, can be constructed based on pairing on elliptic curves [Jou00, BF01] and have been used in cryptographic literature for over 15 years, the first obfuscation candidates required *polynomial* level (in the "security parameter" of the scheme). Proposed constructions of multilinear maps for level > 2 have only started to emerge recently [GGH13a, CLT13, CLT15, GGH15] and their security is highly questionable. Indeed, many concrete security assumptions were shown to be broken w.r.t all known candidates with level > 2 [BGH+15, CGH+15, CHL+15, CLR15, HJ15, MF15, CFL+16, CJL16, MSZ16].

A beautiful work of Lin [Lin16a], followed by [LV16, Lin16b, AS16], showed that the required level of multilinearity can be reduced to a constant (ultimately 5 in [Lin16b, AS16]). These works show a relation between the required multilinearity level and the existence of "simple" pseudorandom generators (PRGs). At a rudimentary level, the PRGs are used to "bootstrap" simple obfuscation-like objects into full-fledged obfuscators. This approach requires PRGs mapping $\{0,1\}^n$ to $\{0,1\}^m$ with $m = n^{1+\Omega(1)}$, which can be represented as low-degree polynomials over $\mathbb{R}$.

More accurately, for a security parameter λ and large enough n, the required output length is $m = n^{1+\varepsilon} \cdot \mathrm{poly}(\lambda)$, for some fixed polynomial $\mathrm{poly}(\cdot)$ which is related to the computational complexity of evaluating the underlying cryptographic primitives. One can ensure this condition as long as the output length is at least $n^{1+\Omega(1)}$ by setting n to be a sufficiently large polynomial in λ. The situation complicates further when trying to optimize the concrete constant corresponding to the level of multilinearity by means of preprocessing as in [Lin16b, AS16, LT17]. The stretch bound needs to hold even with respect to the preprocessed seed length (see Appendix B for more details).

Lin [Lin16b] and Ananth and Sahai [AS16] instantiated this approach with locality-5 PRGs, which can trivially be represented as degree 5 polynomials. Their main insight was that for constant locality PRGs, preprocessing only blows up the seed by a constant factor. However, even so, the required stretch is impossible to achieve with locality smaller than 5 [MST06].

Implications of our Work to Candidate Bilinear-Maps-Based Constructions. Very recently, Lin and Tessaro [LT17] proposed an approach to overcome the locality barrier and possibly get all the way to an instantiation of iO based on bilinear maps. This could be a major breakthrough in cryptographic research, allowing to base "fantasy" cryptography on well studied hardness assumptions. Lin and Tessaro showed that it is sufficient if the PRG has low *block-wise locality* for blocks of logarithmic size. Namely, if we consider the seed of the PRG as an $b \times n$ matrix for $b = O(\log n)$, then each output bit can be allowed to depend on ℓ columns of this matrix. The required output length is $m = 2^{c \cdot b} n^{1+\Omega(1)}$ for some constant c. An explicit value for c is not given, but the construction requires $c > 3$ which seems to be essential for this approach (see Appendix B).Block-wise locality allows a possible way to bypass the impossibility results for standard (i.e., bitwise) locality, and indeed Lin and Tessaro conjectured that there is a pseudorandom generator with output length $n^{1+\Omega(1)}$ and block-wise locality $\ell = 2$, and proposed a candidate construction.

Theorem 1.3 shows that generators with block-wise locality 2 cannot have the stretch required by the [LT17] construction, thus suggesting that their current techniques are insufficient for achieving obfuscation from bilinear maps. While our worst-case result leaves a narrow margin for possible

improvement of the obfuscation reduction to work with $1 < c < 2$, our improved analysis for random graphs and predicates (see Theorem 5.16 in Section 5.3) suggests that our methods may be effective, at least heuristically, for generators with *any* $c > 1$.

Ananth et al. [ABKS17] observed that there is a way to generalize the [LT17] approach, so that it is sufficient that the range of the PRG is not $\{0,1\}$, but rather some small specified set, so long as the degree (as a polynomial over the rationals) is bounded by the level of multilinearity. Furthermore, pseudorandomness was no longer a requirement, but rather it is only required that the output of the generator is indistinguishable from some product distribution (in particular, the one where each output entry is distributed according to its marginal). This suggests that perhaps a broader class of generators than ones that have been considered in the literature so far are useful for reducing the degree of multilinearity. However, their approach imposes a number of restrictions on such generators in order to be effective. In particular, it requires preprocessing which increases the seed length by a factor of s^c, for some $c > 1$, where s is the number of monomials in each output coordinate of the generator. Therefore, Theorem 1.5 rules out the applicability of this technique for degree 2 generators, as well.

Supporting Evidence for Block-Wise Locality 3. We show that while the Lin-Tessaro approach might not yet bring us all the way to level 2, it is quite plausible that it implies a construction from tri-linear maps. Namely, that any improvement on the state of the art would imply full-fledged program obfuscators. Specifically, as explained in Section 1.1, we present a candidate generator of block-wise locality 3, with *constant* size blocks. We show that this candidate is robust against algorithms such as ours, as well as other algorithmic methods. See Section 7 for more details.

3 Our techniques

In this section we give an informal overview of the proof of our main result, Theorem 1.5 (i.e., limitations of low degree generators), focusing mostly on the degree two case, and making some simplifying assumptions. For the full proof see Section 4. We also describe at a high level, the ideas involved in the improved algorithm for the special cases of single-predicate generators (Theorem 1.4), random block-local generators (Theorem 5.16) and sum-of-squares lower bound (Theorem 1.6) that shows a generator with stretch $m = \Omega(n2^b)$ that is resistant to sum-of-squares based attacks (an algorithm that encapsulates all our techniques.)

As we observe in Section 3.1 below, Theorem 1.5 can be used in a black-box way to obtain a slightly weaker variant of Theorem 1.3, showing limitations of two block-local (and more generally ℓ block-local) generators. The full proof of Theorem 1.3, with the stated parameters, appears in Section 5.

Our work builds on some of the prior tools used for analyzing local pseudorandom generators and refuting constraint satisfaction problems, and in particular relies on *semidefinite programming*. The key technical difference is that while prior work mostly focused on generators/predicates with *constant* input locality or arity, we consider functions that could have much larger input locality, but have small degree. The fact that (due to our motivations in the context of obfuscation) we consider mappings with *non-Boolean* output also induces an extra layer of complexity.

We now describe our results in more detail. For simplicity, we focus on the degree two case, which is the case that is of greatest interest in the application for obfuscation. Recall that a *degree-two map of* $\mathbb{R}^n$ *to* $\mathbb{R}^m$ is a tuple of m degree two polynomials $\bar{p} = (p_1, \ldots, p_m)$. We will assume that the

polynomials are *normalized* in the sense that $\mathbb{E}\, p_i(U) = 0$ and $\mathbb{E}\, p_i(U)^2 = 1$ for every i. Let Z be some "nice" (e.g., $O(1)$-spread) distribution over $\mathbb{R}^m$. (For starters, one can think of the case that Z is the uniform distribution over $\{\pm 1\}^n$, though we will want to consider more general cases as well.) The *image refutation problem* for the map $\bar{p}$ and the distribution Z is the task of certifying, given a random element z from Z, that $z \notin \bar{p}(\{\pm 1\}^n)$.

A natural approach is to use an approximation or refutation algorithm for the constraint satisfaction problem obtained from the constraints $\{p_i(x) = z_i\}$ for every i. The problem in our case is that while each of these predicates is "simple" in the sense of having quadratic degree, it can have very large locality or arity. In particular, the locality can be as large as s— the number of monomials of p_i— which we typically think of as equal to n^ε for some small $\varepsilon > 0$.

Much of the CSP refutation literature (e.g., see [AOW15a]) followed the so called "XOR principle" which reduces the task of refuting a CSP with arbitrary predicates, to the task of refuting a CSP where all constraints involve XORs (or products, when the input is thought of as ± 1 valued) of the input variables. Generally, applying this principle to arity s predicates leads to a 2^s multiplicative loss in the number of constraints, and also yields XORs that can involve up to s variables, which is unacceptable in our setting. However, as shown by [AOW15a], the situation is much better when the original predicate has small degree d (which, in particular, means it does not support a $(d+1)$-wise-independent distribution). In this case, utilizing the XOR principle results in a d-XOR instance, and only yields roughly an s^d loss in the number of constraints.

However, there are two issues with this approach. First, this reduction is not directly applicable in the non-Boolean setting, which is relevant to potential applications in obfuscation. Second, reducing to an XOR inherently leads to a loss in the output length that is related to the sparsity s, while, as we'll see, it may be sometimes possible to avoid losing such factors altogether.

Thus, our algorithm takes a somewhat different approach. Given the variables $z_1, \ldots, z_m$, we consider the quadratic program

$$\max_{x \in \{\pm 1\}^n} \sum_{i=1}^{m} z_i p_i(x) \,. \tag{3.1}$$

The value of this program can be approximated to within a $O(\log n)$ factor using semidefinite relaxation via the *symmetric Grothendieck inequality* of Charikar and Wirth [CW04]. Thus, it is sufficient to show a gap in the value of this program between the "planted" case, where there is some x such that $p_i(x) = z_i$ for every i, and the case where the values z_i are sampled from Z.

If there is some x such that $p_i(x) = z_i$ for every i, then the value of the program (3.1) is at least $\sum_{i=1}^m z_i^2$ which (using the fact that $\mathbb{E}\, z_i^2 = 1$ and standard concentration bounds) we can assume to be very close to m.[8]

On the other hand, consider the case where $(z_1, \ldots, z_m)$ is chosen from Z. For every fixed $x \in \{\pm 1\}^n$, we can define m random variables $Y_1^x, \ldots, Y_m^x$ such that $Y_i^x = z_i p_i(x)$ and let $Y^x = \sum_{i=1}^m Y_i^x$. Since Z is a product distribution, the random variables Y_i^x are independent, and hence we can use the Chernoff bound to show that with all but $0.01 \cdot 2^{-n}$ probability, the value of Y^x will be at most $O(\sqrt{nBm})$, where B is a bound on the magnitude of $z_i p_i(x)$. We can then apply the union bound over all possible x's to show that the value of the quadratic program (3.1) is at most $O(\sqrt{nBm})$ with probability 0.99.

For example, if each z_i is a uniform element in $\{\pm 1\}$, and $|p_i(x)| \leqslant O(1)$ for every x (as is the case when p_i is a *predicate*), then $B = O(1)$ and so in this case the value of (3.1) will be at most

[8] Formally, in the case that $p_i(x) = z_i$ we do not assume anything about the distribution of z. However, if $\sum_{i=1}^m z_i^2 < 0.9m$, we can simply choose to output "?".

m/c as long as $m \gg c^2 n$. Setting c to the aforementioned approximation factor $O(\log n)$, we get a successful refutation.

The resulting algorithm does the following. On input $z_1, \ldots, z_m$, run the SDP relaxation for (3.1) and if the value is smaller than $m/2$, then output "`refuted`" and declare that z is not in the image of G. In the case where $z = \mathcal{G}(x)$ the value of the quadratic program, and so also its SDP relaxation, will be at least $0.9m$.[9] On the other hand, if $m = \omega(n \log n)$, then with high probability the value of the quadratic program will be $o(m/\log n)$ and hence the relaxation will have value $o(m)$.

In the discussion above we made two key assumptions:

- $|p_i(x)| \leqslant O(1)$ for every $x \in \{\pm 1\}^n$
- $|z_i| \leqslant O(1)$ for $x \in \{\pm 1\}^n$

In general both of these might be false. If p_i has at most s non-zero monomials, and satisfies $\mathbb{E}\, p_i(U)^2 = 1$, then we can show that $|p_i(x)| \leqslant \sqrt{s}$ for every x, using the known relations between the ℓ_1 and ℓ_2 norms of p_i's Fourier transform. The second condition can be a little more tricky. If the z_i's are *subgaussian*, then we can use Hoeffding's inequality in place of the Chernoff bound, but in general we cannot assume that this is the case. Luckily, it turns out that in our application we can use a simple trick of rejecting outputs in which z_i has unusually large magnitude to reduce to the bounded case. The bottom line is that we get an efficient algorithm for the image-refutation problem of an s-sparse quadratic map whenever $m \gg sn \log n$.

The higher degree case reduces to the degree 2 by "quadratisizing" polynomials. That is, we can consider a degree d polynomial on n variables as a degree 2 polynomial on the $n^{\lceil d/2 \rceil}$ variables obtained by considering all degree $\lceil d/2 \rceil$ monomials. Using this approach, we can generalize our results (at a corresponding loss in the bound on the output) to higher degree maps.

3.1 Distinguishing generators with block-locality 2

A priori the notions of *block locality* and *algebraic degree* seem unrelated to one another. After all, a two block local generator on size b blocks could have degree that is as large as $2b$. However, we can *pre-process* a length bn input $x \in \{\pm 1\}^{bn}$, by mapping it to an input $x' \in \{\pm 1\}^{n'}$ for $n' = 2^b n$ where for every $i \in [n]$, the i^{th} block of x' will consist of the values of all the 2^b monomials on the i^{th} block of x. Note that a map of block locality ℓ in x becomes a map of *degree* ℓ in x'. Moreover, since every output bit depends on at most ℓ blocks, each containing 2^b variables, the number of monomials in this degree ℓ polynomial is at most $2^{\ell b}$.

In this way, we can transform a candidate two block-local pseudorandom generator $\mathcal{G}\colon \{\pm 1\}^{bn} \to \{\pm 1\}^m$ into a degree-2 sparsity-2^{2b} map $\mathcal{G}'\colon \{\pm 1\}^{n'} \to \mathbb{R}^m$. Note that even if $\mathcal{G}$ is a secure pseudorandom generator, it is *not* necessarily the case that $\mathcal{G}'$ is also a pseudorandom generator, as the uniform distribution on $x \in \{\pm 1\}^{bn}$ does not translate to the uniform distribution over $x' \in \{\pm 1\}^{2^b n}$. However, the image of $\mathcal{G}'$ contains the image of $\mathcal{G}$, and hence if we can solve the image refutation problem for $\mathcal{G}'$, then we can do so for $\mathcal{G}$ as well. Applying the above result as a black-box gives an efficient algorithm to break a two block-local generator of block size b as long as the output length m satisfies

$$m \gg 2^{2b} n' \log^2 n = 2^{3b} n \log^2 n \ .$$

[9] We ignore here the case where $\sum z_i^2 < 0.9m$, in which case our algorithm will halt with the output "?".

This is already enough to break the concrete candidate of Lin and Tessaro [LT17], but a more refined analysis shows that we can improve the 2^{3b} factor to 2^{2b}. Furthermore, if we initialize the construction with a random predicate on an expanding constraint graph we can bring this factor down to 2^b. Both improvements still use the same algorithm, only providing a tighter analysis of it in these cases. We do not know if our analysis can be improved even further. Mapping out the various trade-offs for block-local generators (or, equivalently, refuting very large alphabet CSPs), is a very interesting open question.

The first improvement, described in Section 5.1, yields a better bound on the output of any two-block-wise generator. As mentioned above, it uses the same algorithm. That is, we take a candidate two-block-local generator $\mathcal{G}\colon \{\pm 1\}^{bn} \to \{\pm 1\}^m$ and transform it into a degree two mapping $\mathcal{G}'\colon \{\pm 1\}^{2^b n} \to \mathbb{R}^m$ by "expanding out" the monomials in each block. We then run the same algorithm as before on the generator $\mathcal{G}'$, but the key idea is that because $\mathcal{G}'$ arose out of the expansion of a two-block-local generator, we can show a better upper bound on the objective value of the quadratic program (3.1). Specifically, we can express each of these polynomials as a function of the Fourier transform of the predicate that the original block local generator applied to each pair of blocks. We can then change the order of summations, which enables us to reduce bounding (3.1) to bounding 2^{2b} "simpler" sums, for which we able to obtain, in the random case, tighter bounds with sufficiently high probability that allows to take a union bound over these 2^{2b} options. See Section 5.1 for the full detail.

3.2 Improving the Stretch to $n2^b$ for the Single Predicate Case

The second improvement (Theorem 5.3), considers the special case where each output of the generator is computed using the same predicate (as discussed before, this case is the principle focus of [LV17b]). In this case, we show that our image refutation algorithm works whenever m (the number of outputs) of the generator satisfies $m = \tilde{\Omega}(n2^b)$. This matches the stretch required for the *distinguisher* of [LV17b] to work.

We now describe at a high level, how our refutation algorithm works. The refutation algorithm is given a string $z \in \{\pm 1\}^m$ and description of the generator $\mathcal{G}$ that includes the underlying graph G on n vertices and the predicate $P\colon \{\pm 1\}^b \times \{\pm 1\}^b \to \{\pm 1\}$. As a first step, we will reduce the problem of image refuting $\mathcal{G}$ to image refuting a somewhat simpler $\mathcal{G}'$ where the predicate P will be replaced by a "product-predicate" P'. A predicate $P'\colon [q] \times [q] \to \{\pm 1\}$ is a *product* predicate if it can be written as a product of two functions $f\colon [q] \to \{\pm 1\}$ and $g\colon [q] \to \{\pm 1\}$ applied to each of the inputs to P. In the second step, we will give an efficient algorithm for image-refuting two-block-local, single product predicate PRG.

We now describe the first step. Here, the algorithm wishes to certify that there's no $x \in (\{\pm 1\}^b)^n$ such that $\mathcal{G}(x) = z$. Fix any $x \in (\{\pm 1\}^b)^n$. For this fixed x, consider the distribution $\mathcal{D}$ on inputs to P, generated by taking a random edge $\{i, j\}$ in G and outputting (x_i, x_j). We will show, using a result of Linial and Schraibman shown in the context of relating marginal complexity to various measures of communication complexity, that on $\mathcal{D}$ (more generally, any distribution on inputs to P), there's a product predicate $F(\alpha, \beta) = f(\alpha) \cdot g(\beta)$ such that $\mathbb{E}_{(\alpha,\beta)\sim\mathcal{D}}[P(\alpha, \beta) \cdot F(\alpha, \beta)] \geqslant \Theta(2^{-b/2})$. Thus, if there is an $x \in (\{\pm 1\}^b)^n$ such that $\mathcal{G}(x) = z$, then for the same x, $\mathbb{E}_{i\sim[m]}[\mathcal{G}'(x)_i \cdot z_i] \geqslant \Theta(2^{-b/2})$. If we can now certify an upper bound of $\ll 2^{-b/2}$ on $\mathbb{E}_{i\sim[m]}[\mathcal{G}'(x)_i \cdot z_i]$ for every x and with high probability over the draw of z, we'd obtain an image refutation algorithm. This latter question turns out to be simpler because of the product nature of the predicate defining $\mathcal{G}'$.

This step in our algorithm is inspired by the use of a result of Chor-Goldreich in the work of

[LV17b]. This lemma says[10] that for the uniform distribution on the inputs to P, there's a product predicate that has a correlation of $\Theta(2^{-b/2})$ with P. In the work of [LV17b] this observation is used to replace P by a *constant-alphabet* predicate (obtained by massaging the constituents of the product predicate given by Chor-Goldreich lemma above) to obtain a simplified PRG on constant-alphabet size such that when the seed is chosen according to the uniform distribution on $(\{\pm 1\}^b)^n$, the modified PRG's output distribution correlates well with that of the original one. Thus, a strong enough refutation algorithm (they use one due to [AOW15a]) applied to the modified PRG is enough to give a distinguisher. Observe that this approach doesn't give a refutation algorithm because the key step of replacing P with $f \cdot g$ relies on x being drawn uniformly from $[q]^n$.

Instead of using off-the-shelf refutation algorithms (such as that of [AOW15b]), we solve the image refutation problem for single product predicate block-local PRGs by giving a direct, simple algorithm – this algorithm crucially works without the knowledge of the product predicate itself or even the block size parameter b. This is important, as our argument that obtains $\mathcal{G}'$ is not constructive, in particular, the distribution that the product predicate approximates P on is a complicated function of the (purported) arbitrary assignment x and the graph G. Thus, our product-predicate refutation algorithm must work without the explicit knowledge of the underlying product predicate.

Indeed, we show (in Lemma 5.7) that given a graph G on n vertices with $m \gg n$ edges and any string z, we can (in one shot) show that z (w.h.p) is not in the image of *any* of the (infinitely many!) generators obtained by using any two-block-local product predicate of arbitrarily large block size with the same underlying graph G. In particular, our refutation algorithm does not need to know the predicate itself or even the number of bits in each block of the seed for the generator!

3.3 Random Block Local Generators

We analyze the natural candidate of multiple-predicate, block-local generators, where both the underlying graph and each of the predicates are chosen uniformly at random (conditioned on the predicates being balanced), and show (see Section 5.3) that our refutation algorithm works whenever $m = \Omega(n2^b)$. As before, our idea to consider the problem of maximizing the polynomial $\sum_i z_i p_i(x)$. We work with the *matrix* M such that our target polynomial $\sum_i z_i p_i(x)$ is a bilinear form of M. To obtain a certificate for the upper bound on the polynomial, it then suffices to show a strong enough upper bound on the *spectral* norm of the matrix M – which we show is small enough (w.h.p) because of the randomness involved in defining the generator. M has some dependencies between its various entries that preclude the use of standard bounds to upper bound the spectral norm. So we compute an upper bound on the spectral norm using the standard trace method that reduces the problem to some combinatorial properties that are simple to reason about.

4 Image refutation for low degree maps

In this section we will prove our main technical theorem, which is an algorithm for the image refutation problem for every low degree map and "nice" or "non-degenerate" product distributions. We start by defining the notion of non-degenerate distributions, which amounts to distributions

[10] We use a somewhat different way to describe the use Chor-Goldreich lemma by [LV17b] in order to show how it inspires our approach.

that do not put almost all their probability mass on a small (compared to their standard deviation) interval.

Definition 4.1 (c-spread distributions)**.** Let Z be a product distribution over $\mathbb{R}^m$ with $\mathbb{E} Z_i = 0$ and $\mathbb{E} Z_i^2 = 1$ for every i. We say that Z is *c-spread* if for every interval $I \subseteq \mathbb{R}$ of length $1/c$, the probability that $Z_i \in I$ is at most 0.9.

Normalized low-degree maps are polynomials over $\{\pm 1\}^n$ - we use the standard Fourier basis (e.g., see [O'D14]) to represent them:

Definition 4.2 (Fourier notation)**.** For any $S \subseteq [n]$, let $\chi_S(x) = \Pi_{i \in S} x_i$ for any $x \in \{\pm 1\}^n$. A function $p \colon \{\pm 1\}^n \to \mathbb{R}$ can be uniquely expanded as $\sum_{S \subseteq [n]} \hat{p}(S)\chi_S$ where the "Fourier coefficients" $\hat{p}(S) = \mathbb{E}_{x \sim \{\pm 1\}^n}[\chi_S(x)p(x)]$ and the expectation is over the uniform distribution over the hypercube $\{\pm 1\}^n$. Fourier coefficients satisfy the Parseval's theorem: $\mathbb{E}_{x \sim \{\pm 1\}^n} p(x)^2 = \sum_{S \subseteq [n]} \hat{p}(S)^2$.

We define a normalized degree d map to be a collection of degree d polynomials $\bar{p} = (p_1, \ldots, p_m)$ mapping $\{\pm 1\}^n$ to $\mathbb{R}^m$ such that $\mathbb{E}\, p_i(U) = 0$ and $\mathbb{E}\, p_i(U)^2 = 1$ for every i where U is the uniform distribution.[11]

Our main technical theorem is the following:

Theorem 4.3 (Main theorem)**.** *There is an efficient algorithm that solves the refutation problem for every normalized degree d map $\bar{p}$ and c-spread probability distribution Z as long as*

$$m > K \cdot c^2 s(\bar{p}) n^{\lceil d/2 \rceil} \log^2(n) \tag{4.1}$$

for some global constant K.

To state the result in a stronger form, we use a somewhat technical definition for the parameter $s(\bar{p})$, which is deferred till later (see Equation (4.5) and Definition 4.9 below). However, one important property of it is that for every normalized polynomial map $\bar{p} = (p_1, \ldots, p_m)$, $s(\bar{p})$ is smaller than the maximum *sparsity* (i.e., number of monomials) of the polynomials. Hence, Theorem 4.3 implies Theorem 1.5 from Section 1.1. The fact that we only require a factor of $s(\bar{p})$ as opposed to the sparsity makes our result stronger, and in some cases this difference can be very significant.

The algorithm for proving Theorem 4.3 is fairly simple:

[11] Note that we are using the same normalization for the Z_i's and $p_i(U)$, which makes sense in the context of a pseudorandom generator applied to the uniform distribution over the seed. If we wanted to consider other distributions D over the seed, we would need to require that $\mathbb{E}\, p_i(D)^2$ is not much smaller than $\mathbb{E}\, p_i(U)^2$. This condition is satisfied by many natural distributions.

Refutation algorithm
Input: $z \in \mathbb{R}^m$, $p_1, \ldots, p_m$ normalized polynomials of degree d in $\{\pm 1\}^n$.
Output: "`refuted`" or "?".
Operation:

1. Let $I = \{i \in [m] : z_i^2 \leqslant 100\}$. Let μ_i be the conditional expectation of z_i conditioned on $z_i^2 \leqslant 100$.
2. If $\sum_{i \in I}(z_i - \mu_i)^2 < m/(10c)$ return "?".
3. Let θ be the value of the degree $\lceil d/2 \rceil$ SOS relaxation for the degree d polynomial optimization problem
$$\max_{x \in \{\pm 1\}^n} \sum_{i \in I}(z_i - \mu_i)p_i(x) \tag{4.2}$$
4. Return "`refuted`" if $\theta - \sum_{i \in I} \mu_i(z_i - \mu_i) < m/(10c)$ otherwise return "?".

The *degree d sum of squares program* is a semidefinite programming relaxation to a polynomial optimization problem, which means that the value θ is always an upper bound on (4.2). The most important fact we will use about this program is the *symmetric Grothendieck Inequality* of Charikar and Wirth [CW04], which states that in the important case where $d = 2$, the *integrality gap* of this program (i.e., ratio between its value and the true maximum) is $O(\log n)$.

For this case, where $d = 2$, this program is equivalent to the semidefinite program known as the *basic SDP* relaxation for the corresponding quadratic program. This means that θ can also be computed as

$$\max_{\substack{X \in \mathbb{R}^{(n+1)\times(n+1)} \\ X \succeq 0,\ X_{ii}=1\ \forall i}} \operatorname{tr}(A \cdot X)\ , \tag{4.3}$$

where A is an $(n+1) \times (n+1)$ matrix that *represents* the quadratic polynomial $\sum_{i \in I}(z_i - \mu_i)p_i$, in the sense that for every $i, j \in [n]$, $A_{i,j}$ corresponds to the coefficient of $x_i x_j$ in this polynomial, and for every $i \in [n]$, $A_{i,n+1} = A_{n+1,i}$ is the coefficient of x_i.

We now turn to proving Theorem 4.3. We start by showing the case that $d = 2$. The proof for general degree will follow by a reduction to that case.

4.1 Degree 2 image refutation

In this section, we prove Theorem 4.3 for the case $d = 2$, which is restated below as the following lemma:

Lemma 4.4 (Image refutation for degree 2)**.** *There is an efficient algorithm that solves the refutation problem for every normalized degree* 2 *map $\bar{p}$ and c-spread probability distribution Z as long as*

$$m > K \cdot c^2 s(\bar{p}) n \log^2 n \tag{4.4}$$

for some absolute constant $K > 0$.

In this case, the parameter $s(\bar{p})$ is defined as follows:

$$s(p_1, \ldots, p_m) = \tfrac{1}{m} \max_{x \in \{\pm 1\}^n} \sum_{i=1}^{m} p_i(x)^2 \tag{4.5}$$

By expanding each p_i in the Fourier basis as $p_i = \sum \hat{p}_i(S)\chi_S$, we can see that $\max_{x\in\{\pm1\}^n} |p_i(x)| \leqslant \sum |\hat{p}_i|$. Hence, in particular, $s(\bar{p})$ is smaller than the average of the ℓ_1 norm squared of the p_i's Fourier coefficients. Using the fact that $\mathbb{E}\, p_i(U)^2 = 1$, and the standard relations between the ℓ_1 and ℓ_2 norms, we can see that if every one of the p_i polynomials has at most s monomials (i.e., non-zero Fourier coefficients), then $s(\bar{p}) \leqslant s$.

We now prove Lemma 4.4. To do so, we need to show two statements:

- If $z = \bar{p}(x)$, then the algorithm will never output "`refuted`".
- If z is chosen at random from Z, then the algorithm will output "`refuted`" with high probability.

We start with the first and easiest fact, which in fact holds for *every* degree d.

Lemma 4.5. *Let $z \in \mathbb{R}^m$ be such that there exists an x^* such that $p_i(x^*) = z_i$. Then, the algorithm does not output* "`refuted`".

Proof. Suppose otherwise. We can assume that $\sum_{i\in I}(z_i - \mu_i)^2 \geqslant m/(10c)$ as otherwise we will output "?". Since the SDP is a relaxation, in particular, the value θ is larger than $\sum_{i\in I}(z_i - \mu_i)p_i(x^*) = \sum_{i\in I}(z_i - \mu_i)z_i$ under our assumption. Hence, $\theta - \sum_{i\in I}(z_i - \mu_i)\mu_i \geqslant \sum_{i\in I}(z_i - \mu_i)^2 \geqslant m/(10c)$ □

We now turn to the more challenging part, which is to show that the algorithm outputs "`refuted`" with high probability when z is sampled from Z. We start by observing that by Markov's inequality, for every i, the probability that $z_i^2 > 100\,\mathbb{E}\, z_i^2 = 100$ is at most 0.99. Hence, the expected size of the set I defined by the algorithm is at least $0.99m$ and using Chernoff's bound it follows with very high probability that $|I| > 0.9m$. Let Z'_i be the random variable Z_i conditioned on the (probability $\geqslant 0.99$) event that $Z_i^2 \leqslant 100$, and $\mu_i = \mathbb{E}\, Z'_i$. Note that by definition $(Z'_i)^2 \leqslant 100$ with probability 1, i.e. $|Z'_i| \leqslant 10$ with probability 1, which in turn implies that $|\mu_i| \leqslant 10$. By the "spread-out-ness" condition on Z_i and the union bound, $\mathbb{P}[Z'_i \notin [\mu_i - \frac{1}{2c}, \mu_i + \frac{1}{2c}] \geqslant 0.1 - 0.01$ and hence, in particular, $\mathbb{E}[(Z'_i - \mu_i)^2] \geqslant \frac{1}{500c^2}$.

We can consider the process of sampling the z_i values from the algorithm as being obtained by first choosing the set I, and then sampling z_i independently from the random variable Z'_i for every coordinate $i \in I$. The following lemma says that there will not be an *integral* (i.e., $\{\pm1\}$-valued) solution to the SDP with large value.

Lemma 4.6. *With probability at least 0.99 it holds that for every $x \in \{\pm1\}^n$,*

$$\sum_{i\in I}(z'_i - \mu_i)p_i(x) \leqslant O(\sqrt{nms(\bar{p})}) \tag{4.6}$$

Proof. We use the union bound. For every fixed $x \in \{\pm1\}^n$, we let $\alpha_i = p_i(x)$. We know that $\sum_{i\in I}\alpha_i^2 \leqslant \sum_{i=1}^m \alpha_i^2 \leqslant \max_{x\in\{\pm1\}^n}\sum p_i(x)^2 = ms(\bar{p})$. Since $|z'_i - \mu_i| \leqslant 20$, it follows that $(z'_i - \mu_i)$ is sub-gaussian with constant standard deviation. Therefore, $\sum_{i\in I}(z'_i - \mu_i)\alpha_i$ is sub-gaussian with zero expectation standard deviation $O(\sqrt{ms(\bar{p})})$. Therefore, there exists a value $O(\sqrt{nms(\bar{p})})$ s.t. the probability that $\sum_{i\in I}(z'_i - \mu_i)\alpha_i$ exceeds it is smaller than $0.001 \cdot 2^{-n}$. Applying the union bound implies the lemma. □

Lemma 4.4 will follow from Lemma 4.6 using the fact that the SDP gives $O(\log n)$ approximation factor for true maximum. In particular the symmetric version of Grothendieck inequality shown by [CW04] implies that the value θ computed by the algorithm is at most a factor of $O(\log n)$ larger than the true maximum of the integer program (4.2), see Theorem A.3 in Appendix A.

To finish the proof, we need to ensure that (after multiplying by $O(\log n)$) the bound on the RHS of (4.6) will be smaller than $m/(100c) + \sum_{i\in I}(z_i - \mu_i)\mu_i$. Indeed, since $|\mu_i| \leqslant 10$, with high probability over the choice of the z_i's (which are chosen from Z_i'), the quantity $\sum_i (z_i - \mu_i)\mu_i$ is at most, say, 10 times the standard deviation, which is $O(\sqrt{m}) \ll m/c$. (Here no union bound is needed.) So, by plugging in (4.6) what we really need is to ensure that

$$m/(20c \log n) \geqslant O(\sqrt{nms(\bar{p})})$$

or that

$$m \geqslant O(ns(\bar{p})c^2 \log^2 n)$$

which exactly matches the conditions of Lemma 4.4 hence concluding its proof (and hence the proof Theorem 4.3 for the $d = 2$ case).

4.2 Refutation for $d > 2$

In this section, we show how to reduce the general degree d case to the case $d = 2$, hence completing the proof of Theorem 4.3. The main tool we use is the notion of "quadratizing" a polynomial. That is, we can convert a degree d polynomial p on n variables into a degree two polynomial $\tilde{p}$ on $(n+1)^{\lceil d/2 \rceil}$ variables by simply encoding every monomial of degree up to $\lceil d/2 \rceil$ of the input as a separate variable.

Definition 4.7 (Quadratization)**.** Let p be a degree d polynomial on $\mathbb{R}^n$ which we write in Fourier notation (see Definition 4.2) as $p = \sum_{|S|\leqslant d} \hat{p}(S)\chi_S$. Let $d' = \lceil d/2 \rceil$ Then the quadratization of p is the degree two polynomial q on $\binom{n}{\leqslant d'}$ variables defined as:

$$q(y) = \sum_{S,T} \hat{p}(S \cup T) y_S y_T$$

where the elements of the $\binom{n}{\leqslant d'}$ dimensional vector y are indexed by sets of size at most d', and this sum is taken over all sets $S, T \subseteq [n]$ of size at most d' such that every element in S is smaller than every element of T, $|S| = \max\{|S \cup T|, d'\}$.

The following simple properties ensured by quadratization are easy to verify:

Lemma 4.8. *Let q be the quadratization of a degree d polynomial p on $\binom{n}{\leqslant d'}$ variables for $d' = \lceil d/2 \rceil$. Then,*

1. *For any $x \in \{\pm 1\}^n$ there exists $y \in \{\pm 1\}^{\binom{n}{\leqslant d'}}$ such that $q(y) = p(x)$.*
2. $\sum_{S,S'} \hat{q}(\{S, S'\})^2 = \sum_T \hat{p}(T)^2$.
3. $\max_{y\in\{\pm 1\}^{\binom{n}{\leqslant d'}}} q(y) \leqslant \sum_{|T|\leqslant d} |\hat{p}(T)|$.

Proof sketch. For 1, we let $y_S = \chi_S(x)$ for every $|S| \leqslant d'$. For 2 and 3, we note that the set of nonzero Fourier coefficients of p and q is identical because for every set $|U| \leqslant d$ there is a unique way to split it into disjoint sets S, T of size at most d' where S is the first $\min\{|U|, d'\}$ coordinates of U, and $\hat{q}(\{S, T\}) = \hat{U}$. For all other pairs S, T that do not arise in this manner, it will hold that $\hat{q}(\{S, T\}) = 0$. This means that both the ℓ_1 and ℓ_2 norms of the vector $\hat{q}$ are the same as that of the vector $\hat{p}$, implying both 2 and 3. □

We define the complexity of the degree d normalized map $\bar{p}$ as the complexity of the degree 2 normalized map of the quadratizations of p_is:

Definition 4.9 (Complexity of degree d normalized maps)**.** Let $\bar{p}$ be a normalized degree d map and let $\bar{q}$ be its quadratization. Then, we define $s(\bar{p})$ as $s(\bar{q})$ from (4.5).

Remark 4.10. Part 2 of Lemma 4.8 shows that if $\bar{p}$ is normalized the so is its quadratization $\bar{q}$. Part 3 of Lemma 4.8 shows that $s(\bar{p}) \leqslant \text{sparsity}(p)$ for any normalized degree d map p.

We can now complete the proof of Theorem 4.3.

Proof of Theorem 4.3. Let $\bar{p} = (p_1, \ldots, p_m)$ be a normalized degree d polynomial map and let $z_1, \ldots, z_m$ be the inputs given to the algorithm. If there is an x such that $p_i(x) = z_i$ for every i, then by Lemma 4.5 (which did not assume that $d = 2$), the algorithm will return "?".

Suppose otherwise, that $z_1, \ldots, z_m$ are chosen from the distribution Z. Recall that our algorithm computes θ to be the value of the degree $2d'$ SOS relaxation for the quadratic program (4.2). This value satisfies

$$\theta = \max_{\mu(x)} \tilde{\mathbb{E}}_{\mu} \left[\sum_{i \in I} (z_i - \mu_i) p_i(x) \right] ,$$

where the maximum is over all degree $2d'$ pseudo-distributions satisfying $\{x_i^2 = 1\}$ for every $i \leqslant n$.

If μ is a degree $2d'$ pseudodistribution over $\{\pm 1\}^n$ then we can define a degree 2 pseudodistribution μ' over $\{\pm 1\}^{\binom{n}{d'}}$ by having $y \sim \mu$ be defined as $y_S = \chi_S(x)$ for $x \sim \mu$.[12] Let $\bar{q} = (q_1, \ldots, q_m)$ be the quadratization of $\bar{p} = (p_1, \ldots, p_m)$. Then the distribution μ' above demonstrates that $\theta \leqslant \theta'$ where

$$\theta' = \max_{\mu'(y)} \tilde{\mathbb{E}}_{\mu'} \left[\sum_{i \in I} (z_i - \mu_i) q_i(x) \right] .$$

But since this is the value of a degree two SDP relaxation for a quadratic program, we know by Theorem A.3 that it provides an $O(\log n)$ approximation factor, or in other words that

$$\theta' \leqslant O(\log n) \max_{y \in \{\pm 1\}^{\binom{n}{d'}}} \sum_{i \in I} (z_i - \mu_i) q_i(y) . \tag{4.7}$$

Since the q_i's are degree two polynomials over $O(n^{d'})$ variables, Lemma 4.6 implies that when $z_1, \ldots, z_m$ are randomly chosen from Z, w.h.p. the RHS of (4.7) is at most $O((\log n)\sqrt{n^{d'} m s(\bar{q})}) = O((\log n)\sqrt{n^{d'} m s(\bar{p})})$. Setting this to be smaller than $(m/10c^2)$ recovers Theorem 4.3. □

5 Block local generators

Recall that a map $\mathcal{G}\colon \{\pm 1\}^{bn} \to \{\pm 1\}^m$ is ℓ *block-local* if the input can be separated into n blocks of b bits each[13], such that every output of $\mathcal{G}$ depends on at most ℓ blocks.

In this section we will show tighter bounds for block-local generators than those derived from the theorem in Section 4. Of particular interest is the case of block-locality 2 due to its applications for obfuscation from bilinear maps. In Section 5.1 we show a tighter analysis of our algorithm from

[12] While it is clear that this operation makes sense for actual distributions, it turns out to be not hard to verify that it also holds for pseudodistributions, see the lecture notes [BS17].

[13] Our algorithm works even if the blocks intersect arbitrarily. The construction in [LT17] uses only non-intersecting blocks.

Section 4 for any block-local generator. This yields a distinguisher for any block-locality 2 generator with $m \gg 2^{2b} n \log n$. In Section 5.3, we analyze a particularly natural instantiation for 2-block-local PRGs - a random predicate and random constraint graph and show that our distinguisher works for an even smaller $m \gg 2^b n$. In fact, we show that one can even use a simpler distinguisher that computes the largest singular value of a certain matrix arising out of the input instead of running a semidefinite program.

5.1 Bounds on general block-local generators

In this subsection we prove the following result:

Theorem 5.1 (Limitations of block local generators)**.** *For every ℓ-block-local $\mathcal{G}\colon \{\pm 1\}^{bn} \to \{\pm 1\}^m$ there is an efficient algorithm for the $\mathcal{G}$ image refutation problem w.r.t. the uniform distribution over $\{\pm 1\}^m$ as long as*

$$m > (K \log n) 2^{\ell b} (n + 2\ell b)^{\lceil \ell/2 \rceil},$$

where K is a constant depending only on ℓ.

If ℓ is constant and $b = o(n)$ (as is mostly the case), the above translates to refutation for $m > (K \log n) 2^{\ell b} n^{\lceil \ell/2 \rceil}$.

Theorem 1.3 from the introduction is the special case of Theorem 5.1 for the case $\ell = 2$, and so in particular Theorem 5.1 breaks any 2 block local pseudorandom generator with stretch $\tilde{\Omega}(n2^{2b})$ to instantiate the bilinear-map based construction of iO of [LT17].

Remark 5.2. A slightly weaker bound can be obtained by a direct application of Theorem 4.3. We sketch the argument here.

Let $x_1, x_2, \ldots, x_n$ denote elements of $\{\pm 1\}^b$ describing the n input blocks. Let $x_{i,j}$ denote the j^{th} bit of the i^{th} block. For any predicate P, a function of ℓ-blocks, say $x_1, x_2, \ldots, x_\ell$, we can write the Fourier polynomial

$$P(x_1, x_2, \ldots, x_\ell) = \sum_{S_1 \subseteq 1 \times [b], S_2 \subseteq 2 \times [b], \ldots, S_\ell \subseteq \ell \times [b]} \hat{P}(S_1 \cup S_2 \cup \ldots \cup S_\ell) \prod_{i \leqslant \ell} \chi_{S_i}(x_i).$$

Let $x_{i,S} = \chi_S(x_i)$ for any $S \subseteq i \times [b]$ - that is, $x_{i,S}$ is the representation of the b bits of the block in the "Hadamard encoding" as 2^b bits. This encoding of n blocks leads to $n' = n2^b$ variables. Then, the above Fourier polynomial can be equivalently written as

$$P(x_1, x_2, \ldots, x_\ell) = \sum_{S_1 \subseteq 1 \times [b], S_2 \subseteq 2 \times [b], \ldots, S_\ell \subseteq \ell \times [b]} \hat{P}(S_1 \cup S_2 \cup \ldots \cup S_\ell) \prod_{i \leqslant \ell} x_{i,S}.$$

Observe that the degree of P is ℓ in the new variables $x_{i,S}$. Thus, every ℓ-local PRG with m outputs and n inputs is equivalent to a degree ℓ, $\ell 2^b$-arity predicates on $n' = n2^b$ variables. Such polynomials have sparsity at most $2^{\ell b}$.

Applying Theorem 4.3 now yields that if

$$m > K \log(n) 2^{\ell b} n'^{\lceil \ell/2 \rceil} = K \log(n) 2^{\ell b} n^{\lceil \ell/2 \rceil} (2^b \ell)^{\lceil \ell/2 \rceil},$$

then our algorithm solves the image refutation problem establishing that if the output is longer than m, the ℓ-block local PRG is not secure. For $\ell = 2$ in particular, the above analysis yields a threshold of

$$m = K \log(n) 2^{2b} (n 2^b \ell) = K \log(n) 2^{3b} n.$$

As we show next, the analysis of our algorithm can be tightened in the special case when P is a predicate and, in particular, yields that $m > K \log(n) 2^{2b} n$ is enough for our algorithm to establish that 2-block-local PRGs are insecure.

Proof of Theorem 5.1. We begin with a detailed proof for the case of $\ell = 2$. Let G be a graph on n blocks with m edges and let $p_{i,j}$ be a collection of 2-block-local predicates for $(i,j) \in G$ such that each $p_{i,j} \colon [b] \times [b] \to \{0,1\}$ is an arbitrary predicate on $2b$ bits. Let $z \in \{\pm 1\}^m$ be generated uniformly at random – one bit for every edge (i,j) of the graph G. We will certify that there is no $x \in (\{\pm 1\}^b)^n$ such that $p_{i,j}(x_i, x_j) = z_{i,j}$ for every $1 \leqslant i \leqslant m$ with high probability.

For every $(i,j) \in G$, we can write $p_{i,j}(x_i, x_j) = \sum_{S,T \subseteq [b]} \hat{p}_{i,j}(S,T) \chi_S(x_i) \chi_T(x_j)$. We think of this as a degree 2 polynomials $q_{i,j}$ in the $n2^b$ variables $\chi_S(x_i)$ for $S \subseteq [b]$ and $1 \leqslant i \leqslant n$. We run the algorithm from Section 4.1 on the degree 2 polynomials $q_{i,j}$. As outlined above, our analysis in Theorem 4.3 can be used to show that the refutation algorithm succeeds so long as $m = K \log(n) 2^{2b} (n2^b \ell) = K \log(n) 2^{3b} n$. Here, we give a better analysis of the SDP value θ in the algorithm for this special case.

We write:

$$\sum_{(i,j) \in G} z_{i,j} p_{i,j}(x) = \sum_{(i,j) \in G} \sum_{S,T \subseteq [b]} z_{i,j} \cdot \hat{p}_{i,j}(S,T) \chi_S(x_i) \chi_T(x_j).$$

Changing the order of summations, this yields:

$$\sum_{(i,j) \in G} z_{i,j} p_{i,j}(x) = \sum_{S,T \subseteq [b]} \sum_{(i,j) \in G} z_{i,j} \cdot \hat{p}_{i,j}(S,T) \chi_S(x_i) \chi_T(x_j). \tag{5.1}$$

We recall that the SDP relies on the expansion of x into a vector $y \in \{\pm 1\}^{n2^b}$, where $y_{i,S} = \chi_S(x_i)$. Therefore, our SDP relaxation will find $y \in \{\pm 1\}^{n2^b}$ that approximately maximizes the quadratic function $\sum_{S,T \subseteq [b]} \sum_{(i,j) \in G} z_{i,j} \cdot \hat{p}_{i,j}(S,T) y_{i,S} y_{j,T}$. Analogously to Lemma 4.6, we will show that for a large enough constant C it holds that

$$\mathop{\mathbb{P}}_{z \in \{\pm 1\}^m} \left[\max_{y \in \{\pm 1\}^{n2^b}} \sum_{S,T \subseteq [b]} \sum_{(i,j) \in G} z_{i,j} \cdot \hat{p}_{i,j}(S,T) y_{i,S} y_{j,T} > C \sqrt{2^{2b}(n+2b)m} \right] < 0.01 \,, \tag{5.2}$$

which, along with the symmetric Grothendieck inequality, completes the proof analogous to the argument in the proof of Theorem 4.3. To prove that (5.2) holds, we will show that for all $S, T \subseteq [b]$ it holds that

$$\mathop{\mathbb{P}}_{z \in \{\pm 1\}^m} \left[\max_{y \in \{\pm 1\}^{n2^b}} \sum_{(i,j) \in G} z_{i,j} \cdot \hat{p}_{i,j}(S,T) y_{i,S} y_{j,T} > C \sqrt{(n+2b)} \cdot \sqrt{\sum_{(i,j) \in G} |\hat{p}_{i,j}(S,T)|^2} \right] < 0.01 \cdot 2^{-2b} \,. \tag{5.3}$$

Applying the union bound over all 2^{2b} possible values of S, T, together with Cauchy-Schwarz, shows that if (5.3) holds then so does (5.2). To apply Cauchy-Schwarz we recall that $\bar{p}$ are predicates and thus, $\sum_{S,T \subseteq [b]} \hat{p}_{i,j}(S,T)^2 = 1$ for every $(i,j) \in G$. Thus,

$$\sum_{(i,j) \in G} \sum_{S,T \subseteq [b]} |\hat{p}_{i,j}(S,T)|^2 = m \,.$$

Finally, (5.3) holds by concentration. For any fixed S, T, the expression in (5.3) depends on $\{y_{i,S}, y_{j,T}\}_{i,j\in[n]}$ which is a set of at most $2n$ variables (out of the total $n2^b$). Since the z_i's are uniform in $\{\pm 1\}$, by standard Chernoff bounds, it follows that for an appropriate constant C,

$$\mathop{\mathbb{P}}_{z}\left[\sum_{(i,j)\in G} z_{i,j}\cdot \hat{p}_{i,j}(S,T)y_{i,S}y_{j,T} > C\sqrt{(n+2b)}\cdot\sqrt{\sum_{(i,j)\in G}|\hat{p}_{i,j}(S,T)|^2}\right] < 0.01\cdot 2^{-(2n+2b)}\,. \tag{5.4}$$

Applying the union bound over all 2^{2n} possible values of the set $\{y_{i,S}, y_{j,T}\}_{i,j\in[n]}$ implies (5.3) and thus also (5.2).

The case of general ℓ is analogous. We first expand the blocks and obtain $n2^b$ variables. Over the new set of variables, our predicates are functions of degree ℓ. We quadratize them as in the proof of Theorem 4.3 and then apply the argument above to the resulting quadratic polynomials. We omit further details here.

□

5.2 Sharper Bounds on the Stretch of Block-Local PRGs with a Single Predicate

Next, we prove a tighter upper bound of $\tilde{\Theta}(n2^b)$ on the stretch of a block local PRGs with a *single* predicate P (instead of a different predicate for each output) with block-locality 2. The following is the main result of this section:

Theorem 5.3. *For $b \in \mathbb{N}$, let $\mathcal{G}\colon \{\pm 1\}^{bn} \to \{\pm 1\}^m$ be a two block-local PRG defined by an instance graph $G([n], E)$ with $m = |E|$ edges and an arbitrary predicate $P\colon \{\pm 1\}^b \to \{\pm 1\}^b \to \{\pm 1\}$ such that for any seed $x \in (\{\pm 1\}^b)^n$, for every $e \in E$, $\mathcal{G}_e = P(x_{e_1}, x_{e_2})$. Let $z \in \{\pm 1\}^m$.*

Then, for any $m > O(\log^2(n))n2^b$, there exists a $\mathrm{poly}(m, n)$ time algorithm that takes input G, z and P and outputs "`refuted`" or "?" with the following guarantees:

1. *If the output is "`refuted`", then,*

$$\max_{x\in(\{\pm 1\}^b)^n}\sum_{(i,j)\in E} P(x_{e_1}, x_{e_2})z_e < 0.99m.$$

2. *When $z \in \{\pm 1\}^m$ is chosen uniformly at random, then $\mathbb{P}[$ Algorithm outputs "`refuted`"$] > 1 - 1/n$.*

The proof is based on two key observations. The first component shows how to refute a special class of generators that we call *bipartite single-product-predicate* generators with stretch independent of the block-size. The second one shows how to reduce the image refutation problem for arbitrary single-predicate two-block local generators to this special case at the cost of blowing up the stretch by at most a 2^b factor.

5.2.1 Image Refutation for Bipartite Single Product Predicate Generators

We begin by defining the class of *product* predicates and bipartite single-product-predicate generators.

Definition 5.4 (Product Predicates)**.** A two block-local predicate $F\colon \{\pm 1\}^b \times \{\pm 1\}^b \to \{\pm 1\}$ is said to be *product*, if there exist $L\colon \{\pm 1\}^b \to \{\pm 1\}$ and $R\colon \{\pm 1\}^b \to \{\pm 1\}$ such that $F(x, y) = L(x)R(y)$.

Definition 5.5 (Bipartite Single-Product-Predicate Generators)**.** A two-block local, *bipartite, single-product-predicate* generator $\mathcal{G}\colon \{\pm 1\}^{bn} \times \{\pm 1\}^{bn} \to \{\pm 1\}^m$ is described by bipartite instance graph $G([n],[n],E)$ on left and right vertex sets $[n]$, m edges labeling the m outputs and a two-block local product predicate $F\colon \{\pm 1\}^b \to \{\pm 1\}^b \to \{\pm 1\}$ such that for any $x \in \{\pm 1\}^{bn}$ seen as n blocks of b bits each, one for vertex of the graph G, $\mathcal{G}_e = F(x_{e_1}, x_{e_2})$ for every $e \in E$.

A specific example of a bipartite single-product-predicate generator is one where F is a XOR function. That is, $b = 1$ and $L, R\colon \{\pm 1\} \to \{\pm 1\}$ are identity functions. The next simple lemma helps us show that for a given bipartite instance graph $G([n],[n],E)$ with $m = |E|$ edges, refuting the bipartite, single-product-predicate generator $\mathcal{G}$ with the XOR predicate is enough to refute *all* bipartite single-product-predicates with *arbitrary* block lengths!

Lemma 5.6. *Let $G([n],[n],E)$ be a bipartite instance graph with $m = |E|$ edges and $z \in \{\pm 1\}^m$. Suppose there exists a $b \in \mathbb{N}$, a product predicate $F\colon \{\pm 1\}^b \times \{\pm 1\}^b \to \{\pm 1\}$ and $x \in (\{\pm 1\}^b)^n$ such that $\sum_{e\in E} F(x_{e_1}, x_{e_2}) \cdot z_e = m$. Then, there exists an $y \in \{\pm 1\}^{2n}$ such that $\sum_{e\in E} y_{e_1} y_{e_2} z_e = m$.*

Proof. Let $\mathcal{G}$ be the generator with the product predicate F and let F be defined by predicates $L, R\colon \{\pm 1\}^b \to \{\pm 1\}$. Let $\mathcal{G}'$ be generator with the same instance graph G with the product predicate being the XOR predicate (and thus $b = 1$.) Suppose $x \in \{\pm 1\}^{2bn}$ is such that $\mathcal{G}_e = F(x_{e_1}, x_{e_2}) = z_e$. Choose $y \in \{\pm 1\}^{2n}$ by setting $y_{(i,L)} = L(x_i)$ and $y_{(i,R)} = R(x_i)$. It is now easy to verify that $\mathcal{G}'(y) = z$. □

The next lemma is our first tool. It uses the simple observation above (Lemma 5.6) to establish that there's an algorithm that refutes all possible bipartite single-product-predicate generators $\mathcal{G}\colon \{\pm 1\}^{bn} \to \{\pm 1\}^m$ with a given instance graph $G([n],E)$ (for arbitrary values of b). We stress that this algorithm takes only the instance graph as the input and *not* the product predicate or the block size parameter b (and thus, in one shot, refutes all single-product-predicate generators, with all possible block sizes, on the input instance graph).

Lemma 5.7 (Predicate Oblivious Strong Refutation for Single-Product-Predicate Generators)**.** *Let $G([n],E)$ be a directed graph on n vertices and m edges. Let $z \in \{\pm 1\}^m$. For any $\varepsilon > 0$ and $m > 9C_{CW}^2 \log^2(n) n/\varepsilon^2$, there exists an $\mathrm{poly}(m,n)$ time algorithm that outputs "?" or "`refuted`" and satisfies:*

1. *If the output is "`refuted`", then,*

$$\max_{b\in\mathbb{N}} \max_{x\in(\{\pm 1\}^b)^n} \max_{\substack{F\colon \{\pm 1\}^b\times\{\pm 1\}^b\to\{\pm 1\} \\ \text{product predicate}}} \sum_{e\in E} F(x_{e_1}, x_{e_2}) z_e < \varepsilon m.$$

2. *When $z \in \{\pm 1\}^m$ is chosen uniformly at random, then, $\mathbb{P}[$ Algorithm outputs "`refuted`"$] > 1 - 1/n$.*

We will need the following result of Charikar-Wirth here:

Fact 5.8 (Grothendieck's Inequality)**.** *There's a universal constnat C_{CW} such that for any $A \in \{\pm 1\}^{n\times n}$ on n vertices,*

$$\max_{Y\in\mathbb{R}^{n\times n}, Y\succeq 0, Y_{i,i}=1 \text{ for } 1\leqslant i\leqslant n} \sum_{i,j\leqslant n} Y_{i,j} A_{i,j} \leqslant C_{CW} \log(n) \cdot \max_{y\in\{\pm 1\}^n} \sum_{i,j\leqslant n} y_i y_j A_{i,j}.$$

Our algorithm to establish Lemma 5.7 is simple:

Bipartite Product Refutation algorithm
Input: $z \in \{\pm 1\}^m$, bipartite instance graph $G([n],[n],E)$.
Output: "`refuted`" or "?".
Operation:

1. Let $\theta(z)$ be the value of the standard SDP relaxation optimization problem

$$\max_{y \in \{\pm 1\}^{2n}} \sum_{e \in E} y_{e_1} y_{e_2} z_e. \tag{5.5}$$

That is,

$$\theta(z) = \max_{\substack{Y \in R^{2n \times 2n}, \\ Y \succeq 0, \\ Y_{ii}=1 \text{ for } 1 \leqslant i \leqslant 2n}} \sum_{e \in E} Y_{e_1,e_2} z_e. \tag{5.6}$$

2. Return "`refuted`" if $\theta < \varepsilon m$, otherwise return "?".

Proof. The proof is simple and follows from the following two observations.

Claim 5.9. For $m > 9n/\delta^2$, $\mathbb{P}_{z \sim \{\pm 1\}^m}[\max_{y \in \{\pm 1\}^{2n}} \sum_{e \in E} y_{e_1} y_{e_2} z_e > \delta m] \leqslant 1/n$.

Proof of Claim. For a fixed y, the probability that $\max_{y \in \{\pm 1\}^{2n}} \sum_{e \in E} y_{e_1} y_{e_2} z_e > t\sqrt{m}$ is at most $e^{-t^2/2}$. For $t = 3\sqrt{nm}$, this probability is at most e^{-4n}. Using union bound over all possible 2^{2n} values of y yields that $\mathbb{P}_{z \sim \{\pm 1\}^m}[\max_{y \in \{\pm 1\}^{2n}} \sum_{e \in E} y_{e_1} y_{e_2} z_e > 3\sqrt{mn}] \leqslant 1/n$. Finally, if $m > 9n/\delta^2$, then $3\sqrt{mn} \leqslant \delta m$. □

Applying Fact 5.8 with $\delta = \varepsilon / C_{CW} \log(n)$, we immediately obtain the following observation.

Claim 5.10. For $m > 9C_{CW}^2 \log^2(n) n/\varepsilon^2$, $\mathbb{P}_{z \in \{\pm 1\}^m}[\theta(z) < \varepsilon m] \geqslant 1 - 1/n$ where θ is the SDP value computed in (5.6).

We can now use Lemma 5.6 to complete the proof. From Claim 5.10, it is clear that the algorithm outputs "`refuted`" with probability at least $1 - 1/n$. To verify the first property, observe that if there is a $b \in \mathbb{N}$, a product predicate F and an $x \in (\{\pm 1\}^b)^n$ such that $\sum_{e \in E} G_e z_e > \varepsilon m$, then by Lemma 5.6, $\max_{y \in \{\pm 1\}^{2n}} \sum_{e \in E} y_{e_1} y_{e_2} z_e > \varepsilon m$. Since $\theta \geqslant \max_{y \in \{\pm 1\}^{2n}} \sum_{e \in E} y_{e_1} y_{e_2} z_e > \varepsilon m$, the first step in the algorithm never produces a $\theta < \varepsilon m$ and we are done.

□

5.2.2 Image Refutation for Arbitrary Two-Block Local, Single Predicate Generators

In this section, we describe our second component that helps us use the result from the previous section in order to obtain an image refutation algorithm for arbitrary two block local generators with a single predicate with block length b whenever the stretch $m > \tilde{\Theta}(n) \cdot 2^b$.

We start by defining the value of a generator at any given z.

Definition 5.11 (Value)**.** Let $\mathcal{G} \colon \{\pm 1\}^{bn} \to \{\pm 1\}^m$ be defined by an instance graph G and predicate P. Given any $z \in \{\pm 1\}^m$, we write $\mathrm{val}_{\mathcal{G}}(z) = \frac{1}{m} \max_{x \in \{\pm 1\}^{bn}} \sum_{e \in E} \mathcal{G}_e(x) z_e \in [-1, 1]$.

Key to our result is the following claim that shows that one can replace P by a product predicate and still maintain high value.

Lemma 5.12. *Let $\mathcal{G}\colon \{\pm1\}^{bn} \to \{\pm1\}^m$ be defined by an instance graph G and predicate P. Fix $z \in \{\pm1\}^m$ such that $\mathrm{val}_{\mathcal{G}}(z) = 1$. Then, there exists a product predicate F such that $\mathcal{G}'\colon \{\pm1\}^{bn} \to \{\pm1\}^m$ obtained by replacing P by F satisfying: $\mathrm{val}_{\mathcal{G}'}(z) > 2^{-b/2-1}$.*

To show this lemma, we will appeal to a classical result of Linial and Schraibman [LS09].

Fact 5.13 (Lower Bound on Cut-Norm, follows from Lemma 3.3 in [LS09] and Lemma 4.2 in [LMSS07])**.** *Let P be a $q \times q$ matrix with $\{\pm1\}$ entries and rank r. Let μ be a distribution over entries of P, i.e, over $[q]\times[q]$. Then, there's a rank 1 matrix $uv^\top$, $u, v \in \{\pm1\}^q$ such that $\sum_{i,j\leqslant q} P(i,j)\cdot u(i)v(j) \geqslant \frac{1}{2\sqrt{r}}$.*

Proof of Lemma 5.12. Since $\mathrm{val}_{\mathcal{G}}(z) = 1$, there exists an $x \in (\{\pm1\}^b)^n$ such that $\mathcal{G}(x) = z$. For each $(s,t) \in \{\pm1\}^b \times \{\pm1\}^b$, let $n_{s,t} = |\{(x_{e_1}, x_{e_2}) \mid e \in E \text{ and } (x_{e_1}, x_{e_2}) = (s,t)\}|$. Then, $\sum_{s,t\in\{\pm1\}^b} n_{s,t} = m$.

Let μ be the distribution on $\{\pm1\}^b \times \{\pm1\}^b$ defined by the mass function $\mu(s,t) = n_{s,t}/m$ for every $s, t \in \{\pm1\}^b$. Let P (by a slight abuse of notation) be the matrix indexed by $\{\pm1\}^b$ on rows and columns with (s,t)th entry given by $P(s,t)$. We call this the "matrix of the predicate" P.

Apply Fact 5.13 to matrix P and distribution μ and noting that rank of P is at most 2^b, we have that there are functions $u, v\colon \{\pm1\}^b \to \{\pm1\}$ such that

$$\mathop{\mathbb{E}}_{\mu}[P(s,t)u(s)v(t)] \geqslant 2^{-b/2-1}. \tag{5.7}$$

We claim that the product predicate $F(s,t) = u(s)v(t)$ for any $s, t \in \{\pm1\}^b$ satisfies the required conditions. To verify this, we must show that the generator where we replace P by F has value at least $m \cdot 2^{-b/2-1}$. To exhibit a lower bound on the value of this modified generator, we show that its output at the same x as above correlates well with z.

We can estimate this correlation as:

$$\sum_{(i,j)\sim E} F(x_i, x_j)\cdot z_{i,j} = \sum_{(i,j)\sim E} F(x_i,x_j)\cdot P(x_i,x_j) = m \cdot \mathop{\mathbb{E}}_{(s,t)\sim\mu} F(s,t)\cdot P(s,t) = m\cdot 2^{-b/2-1},$$

where in the last equality, we used (5.7). □

Our refutation algorithm will work by reducing to the bipartite single-product-predicate case handled in the previous section. For this reduction, we will need another simple component.

For a two-local generator defined by an instance graph $G([n], E)$, our refutation algorithm will start by "doubling" G to produce an bipartite graph on $2n$ vertices by creating two copies of every block. We define this *doubling* operation next.

Definition 5.14 (Doubling of G, $\mathcal{G}$)**.** Let $G([n], E)$ be a directed instance graph. The doubling of G is a bipartite graph $G'([n], [n], E')$ to be the *bipartite* graph on $|V| = 2n$ vertices produced as follows: for every directed edge $e = (i,j)$ of G, create an bipartite edge in G': (i,j).

The *doubling* of any two block-local generator $\mathcal{G}\colon \{\pm1\}^{bn} \to \{\pm1\}$ with a predicate F is one, that, for any $x \in \{\pm1\}^{2bn}$ (thought of as a block of b bits for each of the $2n$ vertices of G') outputs m bit string, and for every edge $e = (i,j)$ of G' outputs $\mathcal{G}'_e = F(x_i, x_j)$.

The following is easy to observe:

Lemma 5.15. *Suppose $z \in \{\pm1\}^m$ is such that $\mathcal{G}(x) = z$ for some $x \in \{\pm1\}^{bn}$ where $\mathcal{G}$ is a two block-local generator $\mathcal{G}\colon \{\pm1\}^{bn} \to \{\pm1\}^m$. Then, there exists a $y \in \{\pm1\}^{2bn}$ such that $\mathcal{G}'(y) = z$ for $\mathcal{G}'$, the doubling of $\mathcal{G}$.*

Proof. Identify the vertices of $\mathcal{G}'$ with (i, L) and (i, R) for $i \in [n]$ (where L and R stand for "left" and "right", respectively). Define $y \in \{\pm 1\}^{2bn}$ so that $y_{(i,L)} = y_{(i,R)} = x_i$. It's easy to verify that $\mathcal{G}'(y) = z$. □

We can now complete the proof of Theorem 5.3.

Proof of Theorem 5.3. Let $\mathcal{G}' \colon \{\pm 1\}^{bn} \times \{\pm 1\}^{bn} \to \{\pm 1\}^m$ be the generator with instance graph $G'([2n], E')$ obtained by following two transformations in sequence.

1. Replace P by the product predicate given by Lemma 5.12.
2. Applying doubling.

Then, $\mathcal{G}'$ is a bipartite single-product-predicate generator. We apply the algorithm from Lemma 5.7 to $\mathcal{G}'$ for $\varepsilon = 2^{-b/2-1}$.

Thus, if $m > 9(2n)C_{CW}^2 \log^2(n) 2^{b+2} = 72C_{CW}^2 \log^2(n) n 2^b$, then the algorithm outputs "`refuted`" with probability at least $1 - 1/n$. Further, observe that if there is an $x \in (\{\pm 1\}^b)^n$ such that $\mathcal{G}(x) = z$, then, applying Lemma 5.15 and Lemma 5.12, there's an x' such that $\sum_{e \in E'} \mathcal{G}'(x')_e z_e > 2^{-b/2-1} m$. On the other hand, by Lemma 5.7, we know that whenever the algorithm outputs "`refuted`", the instance $\mathrm{val}'_{\mathcal{G}}(z) \leqslant \varepsilon m = 2^{-b/2-1} m$. Thus, the first condition of the theorem is satisfied. □

5.3 Image Refutation for Random Block-Local PRGs

A particularly appealing construction of block local PRGs is obtained by instantiating them with a random graph with $\sim m$ edges and a random and independent predicate for every edge. A priori, the randomness in this construction could appear to *aid* the security of the PRG. Indeed, such instantiations are in fact suggested by [LT17]. We show that in this case, as in the previous section where all predicates are identical, we can show a *stronger* upper bound on the stretch of the local PRG in terms of the block size b. Whereas in Section 5.1, for general block-local PRGs with non-identical predicates, we lost a factor of $2^{2b} \log(n)$ in the output length, for the special case of a random graphs and random, independent predicates, this can be improved to $\Theta(2^b)$ as we show in this section. We note that the only property of random graphs that we use is expansion.

More concretely, in this section, we analyze the stretch of the following candidate construction of a block-local PRG.

- We choose a graph $G([n], q)$ where every edge is present in G with probability $q = \frac{m}{\binom{n}{2}}$. Thus, with high probability, the number of edges in the graph is $m \pm \sqrt{m}$.
- For every edge $\{i, j\}$ in G, we choose a uniformly random predicate $P_{i,j}(x, y) = \pm 1$ conditioned on $P_{i,j}$s being balanced, i.e. $\mathbb{E}_{x,y \sim \{\pm 1\}^b} P_{i,j}(x, y) = 0$.
- On input (seed) $x \in \{\pm 1\}^{bn}$, which we think of as partitioned into blocks $x_1, \ldots, x_n \in \{\pm 1\}^b$, the generator outputs $h_{i,j}(x_i, x_j)$ for every edge (i, j) of G.

Theorem 5.16 (Limitations of random block-local generators)**.** *There is some constant K such that if $\mathcal{G} \colon \{\pm 1\}^{bn} \to \{\pm 1\}^m$ is a generator sampled according to the above model and $m \geqslant K 2^b n \log^3(n)$, then w.h.p. there is a polynomial-time algorithm for the $\mathcal{G}$ image refutation problem w.r.t. the uniform distribution over $\{\pm 1\}^m$.*

Proof. We identify the underlying graph G with its collection of edges, and hence can think of the outputs of the pseudorandom generator $\mathcal{G}$ as indexed by $(i,j) \in G$. For every edge i,j in G, we think of the predicate $P_{i,j}$ as a $2^b \times 2^b$ matrix indexed by strings in $\{\pm 1\}^b$ that is, for any $\alpha, \beta \in \{\pm 1\}^b$, the (α,β)-entry of the matrix $P_{i,j}$ is given by $P_{i,j}(\alpha,\beta)$.

For any $x = (x_1, x_2, \ldots, x_n)$, with $x_i \in \{\pm 1\}^b$, define $y = y(x) \in \{0,1\}^{2^b n}$ as a vector indexed by (i,α) for $\alpha \in \{\pm 1\}^b$ and (i,α) entry of y equal to 1 iff $x_i = \alpha$ and 0 otherwise. Further, we see y_i as a vector in 2^b dimensions indexed by $\alpha \in \{\pm 1\}^b$ itself. Then, as before, for any $z \in \{\pm 1\}^G$, the fraction of constraints satisfied by any assignment $x \in \{\pm 1\}^{bn}$ is equal to $\max_{x \in \{\pm 1\}^{bn}} \sum_{\{i,j\} \in G} z_{i,j} P_{i,j}(y(x))$.

As in the previous sections, our approach for the image refutation problem will involve showing that the term

$$\tau(z) = \max_{x \in \{\pm 1\}^{2^b n}} \sum_{(i,j) \in G}^{m} z_{i,j} P_{i,j}(y(x)) \tag{5.8}$$

takes very different values in the planted case (where $z = \mathcal{G}(x)$for some x) and in the random case (where z is chosen at random in $\{\pm 1\}^m$), and furthermore proposing an efficient way to approximate $\tau(z)$ that will allow us to distinguish the two cases.

In this section, because of the randomness in the graph G and the predicates $P_{i,j}$s, it actually suffices for us to use the *spectral norm* (i.e., the largest singular value) of an appropriate matrix associated with $\mathcal{G}$ as a certificate of upper bound on $\tau(z)$ instead of semidefinite programming.[14]

Let M be the matrix with rows and columns indexed by (i,α) for $i \in [n]$ and $\alpha \in \{\pm 1\}^b$ and $(i,\alpha),(j,\beta)$-entry of M given by 0 if $\{i,j\}$ is not an edge in G and $z_{i,j} \cdot P_{i,j}(\alpha,\beta)$ otherwise.

We can now describe our algorithm.

The Algorithm. On input $z = (z_{i,j})_{(i,j) \in G}$, our algorithm will consider the $2^b n \times 2^b n$ matrix $M = M(z)$ above.

It will compute the value $\theta = n\|M\|$, where $\|M\|$ is the spectral norm of M. The algorithm will output "?" if $\theta \geqslant m/2$ and "`refuted`" otherwise. (Recall that in our previous algorithms, a similar value θ was computed using a SDP relaxation.)

Let us first establish that θ is in fact an upper bound on $\tau(z)$ as defined in (5.8). Observe that for any x, $y(x)$ is a vector in R^{n2^b} which we think of as indexed by (i,α) for $i \in [n]$ and $\alpha \in \{\pm 1\}^b$. Thus, any i, y_i is a vector of 2^b dimension. Further, y_i is non-zero in exactly one coordinate (namely, the α that equals x_i). As a result, $\|y\|_2^2 = n$ (despite it being a vector of $n2^b$ dimensions). Thus, using (5.8),

$$\tau(z) \leqslant \max_{y \in \{0,1\}^{2^b n} : \|y\|_2^2 = n} \sum_{(i,j) \in G} z_{i,j} P_{i,j}(y) \leqslant \max_{y : \|y\|_2^2 = n} \sum_{i,j} z_{i,j} y_i^\top P_{i,j} y_j = y^\top M y \leqslant \|y\|^2 \cdot \|M\| = n\|M\|, \tag{5.9}$$

To analyze the algorithm's performance, we first notice that in the planted case where $z = \mathcal{G}(x)$, $\theta \geqslant \tau(z) \geqslant m$ since there is always a y, derived from x, for which $z_{i,j} p_{i,j}(y) = 1$ for all $\{i,j\} \in G$. To complete the analysis, it suffices show that when the z_i's are chosen uniformly at random, it will hold with high probability that $\theta < \varepsilon m$ for some sufficiently small $\varepsilon > 0$.

The key component in this is the spectral norm estimate for M show in Lemma 5.17 which proves that $\|M\| \leqslant O(\log^{1.5}(n)) \cdot \sqrt{m/n} 2^{b/2}$. Plugging in this bound yields $\theta \leqslant O(\log^{1.5}(n)) \sqrt{mn} 2^{b/2}$. For

[14] The SOS program is only stronger than this spectral bound so we could have used the exact same algorithm as in Section 4. We use the spectral norm for the sake of clarity of exposition.

$m = K2^b n$, thus $\theta \leqslant O(\log^{1.5}(n))\sqrt{K}n2^b$, which, for a large enough $K = \Theta(\log^3(n))$, yields that $\theta \leqslant m/2$. as required.

□

Lemma 5.17. *Let $M \in \mathbb{R}^{n2^b \times n2^b}$ be the matrix defined above with entries indexed by (i,α) and (j,β) for $i,j \in [n]$ and $\alpha,\beta \in \{\pm 1\}^b$ and $M((i,\alpha),(j,\beta)) = z_{i,j} \cdot p_{i,j}(\alpha,\beta)$ if $\{i,j\} \in G$ and 0 otherwise. Then, with probability at least 0.99, $\|M\| \leqslant O(\log^{1.5}(n)) \cdot \sqrt{\frac{m}{n}}2^{b/2}$.*

Proof. The balancedness constraint on each of the randomly chosen predicates causes some trouble in our analysis - we'd like every non-zero entry of M to be an independent and uniform ± 1 variable. To get to this case, we will define an alternate way of sampling the same $\mathcal{G}$.

We choose the graph in the exact same way as before - each edge included with probability $q = m/\binom{n}{2}$. For each edge in G, say $\{i,j\}$, we sample the predicate $P_{i,j}$ (seen as a $2^b \times 2^b$ matrix as above) in two steps.

1. Sample a matrix $Q_{i,j}$ as follows: sample any entry (α,β) uniformly and independently.
2. Sample $C_{i,j}$ as follows. If $Q_{i,j}$ has $2^{2b-1} + r$ entries equal to $+1$ for $r > 0$, choose r entries of $Q_{i,j}$ that are $+1$ uniformly at random and for those entries, set $C_{i,j}$ to be -2. $C_{i,j}$ has all other entries 0.
3. If $Q_{i,j}$ has $2^{2b-1} + r$ entries that equal -1 for $r > 0$ do the same as above with $+1$ replaced by -1s.

Two easily verifiable facts about $Q_{i,j}$ and $R_{i,j}$ are as follows: 1) the marginal distributions of $Q_{i,j}$ is uniform over all possible sign-matrices of dimension $2^b \times 2^b$ 2) $P_{i,j} = Q_{i,j} + C_{i,j}$ is uniformly distributed over sign matrices with equal number of 1s and -1s.

Let M_1 and M_2 be the matrices whose $(i,\alpha),(j,\beta)$ entries are defined by:

$$M_1((i,\alpha),(j,\beta)) = \begin{cases} 0 & \text{if } \{i,j\} \notin G \\ Q_{i,j}(\alpha,\beta) & \text{otherwise.} \end{cases}$$

$$M_2((i,\alpha),(j,\beta)) = \begin{cases} 0 & \text{if } \{i,j\} \notin G \\ C_{i,j}(\alpha,\beta) & \text{otherwise.} \end{cases}$$

By triangle inequality, $\|M_1 + M_2\| \leqslant \|M_1\| + \|M_2\|$. We will show the following two claims to complete that immediately complete the proof by a union bound.

Claim 5.18. With probability at least 0.999 over the draw from the random model above,

$$\|M_1\| \leqslant \sqrt{m/n}2^{b/2}O(\log(n)).$$

Proof of Claim 5.18. We will use the trace method to analyze the norm of the matrix M_1. Let $N = n2^b$ for notational convenience. The main idea behind the trace method is simple. Let $\sigma_1 \geqslant \sigma_2 \geqslant \ldots \geqslant \sigma_N$ be the singular values of M_1. Then, for any $\ell \in \mathbb{N}$, $(M_1 M_1^\top)^\ell$ has eigen-values $\sigma_1^{2\ell}, \sigma_2^{2\ell}, \ldots, \sigma_N^{2\ell}$. Thus, $\sigma_1^{2\ell} \leqslant \mathrm{tr}((M_1 M_1^\top)^\ell) = \sum_{i=1}^N \sigma_i^{2\ell} \leqslant N\sigma_1^{2\ell}$. In particular, $\mathrm{tr}((M_1 M_1^\top)^\ell)$ gives a factor N approximation to $\sigma_1^{2\ell}$.

Taking expectations, we thus have that $\mathbb{E}[\sigma_1^{2\ell}] \leqslant \mathbb{E}[\mathrm{tr}((M_1 M_1^\top)^\ell)]$. In particular, by Markov's inequality, with probability at least 0.99, $\sigma_1^{2\ell} \leqslant 100\,\mathbb{E}[\mathrm{tr}((M_1 M_1^\top)^\ell)]$. We will choose $\ell = \log(N)$ and upper bound $\mathbb{E}[(\mathrm{tr}((M_1 M_1^\top)^\ell))^{1/2\ell}$ to thus obtain a bound on σ_1 that holds with 0.99 probability.

To analyze $\mathbb{E}[(\mathrm{tr}((M_1 M_1^\top)^\ell))]$, we will expand $\mathrm{tr}((M_1 M_1^\top)^\ell$ as a sum of product of entries of the matrix M_1. We have:

$$\begin{aligned}\mathrm{tr}((M_1 M_1^\top)^\ell &= \sum_{i_1,\alpha_1,i_2,\alpha_2,\ldots,i_{2\ell},\alpha_{2\ell}} M_1((i_1,\alpha_1),(i_2,\alpha_2)) M_1^\top(i_2,\alpha_2,i_3\alpha_3)\cdots M_1(i_{2\ell-1},\alpha_{2\ell-1},i_{2\ell}\alpha_{2\ell}) M_1^\top(i_{2\ell},\alpha_{2\ell},i_1\alpha_1)\\ &= \sum_{i_1,\alpha_1,i_2,\alpha_2,\ldots,i_{2\ell},\alpha_{2\ell}} M_1((i_1,\alpha_1),(i_2,\alpha_2)) M_1(i_3,\alpha_3,i_2\alpha_2)\cdots M_1(i_{2\ell-1},\alpha_{2\ell-1},i_{2\ell}\alpha_{2\ell}) M_1(i_1,\alpha_1,i_{2\ell}\alpha_{2\ell}).\end{aligned}$$

For any (i_j, i_{j+1}) and $\alpha_j\alpha_{j+1}$, observe that $M_1((i_j,\alpha_j),(i_{j+1},\alpha_{j+1})) = G(i_j,i_{j+1}) \cdot P'_{i_j,i_{j+1}}(\alpha_j,\alpha_{j+1})$. Further, random variables G and P' are independent. Thus, taking expectations and using linearity and product rule for expectations yields:

$$\mathbb{E}[\mathrm{tr}((M_1 M_1^\top)^\ell] = \sum_{i_1,\alpha_1,i_2,\alpha_2,\ldots,i_{2\ell},\alpha_{2\ell}} \mathbb{E}[\Pi_{j=1}^{2\ell} G(i_j,i_{j+1})]\,\mathbb{E}[\Pi_{j=1}^{2\ell} P'_{i_j,i_{j+1}}(\alpha_j,\alpha_{j+1})].$$

Let's analyze a term in the summation above. For a term, consider the following parameters: 1) γ: the number of distinct edges of G that appear in the first expectation and 2) for any edge e of G that appears in the first product, let g_e be the number of entries of P_e that appear in the second product.

Now, observe that every non-zero entry of $P'_{i_j,i_{j+1}}$ is mean zero and independent. Thus, if the term contributes a non-zero expectation, then, each distinct P_e must appear an even number of times in the second product. In particular, the number of distinct edges of G that appear cannot be more than $\gamma = 2\ell/2 = \ell$. Next, observe that if there are γ distinct edges of G that appear in the first product, then there are at most $\gamma + 1$ distinct vertices of G that appear in the edges. This can be seen as follows: observe that $i_1, i_2, \ldots, i_{2\ell}, i_1$ forms a cycle. By traversing along the cycle, observe that except the first one, each time a new vertex (one that wasn't encountered before) is traversed, the edge that led to the vertex is also new.

Finally, note that in any non-zero term, the expectation of the second product is exactly 1. The first term has an expectation of $\leqslant (\frac{2m}{n^2})^\gamma$ where γ defined above is the number of distinct edges of G that appear in the first product. We now parameterize the term by the number of distinct vertices $t \leqslant \ell + 1$ (since $\gamma \leqslant \ell$) in a non-zero term and estimate their count: then, the number of distinct ways of choosing t vertices of G is at most n^t. The number of distinct ways of choosing $\geqslant t+1$ edges is at most $t^{2\ell}$. Finally, choosing $(\alpha_i)s$ that have $\ell+1$ distinct elements can be done in at most $\ell^\ell \cdot 2^{b(\ell+1)}$ different ways. The first product for such a term has an expectation of at most $(\frac{2m}{n^2})^{t-1}$. The second term has an expectation of 1.

Thus, we obtain an upper bound of $\ell^{2\ell} \sum_{t\leqslant\ell+1} n^t (\frac{2m}{n^2})^{t-1} 2^{b(\ell+1)} = \ell^{O(\ell)} \cdot m^\ell n^{-\ell-1} 2^{b(\ell+1)}$ on $\mathbb{E}[\mathrm{tr}((M_1 M_1^\top)^\ell]$.

Taking 2ℓth root, this gives an upper estimate on the norm of $\ell \cdot 2^{b(1/2+\frac{1}{2\ell})} \cdot \sqrt{m/n} \cdot n^{\frac{1}{2\ell}}$. By choosing $\ell = \log(n)$, we obtain the claim.

□

Claim 5.19. With probability at least 0.999 over the draw from the random model above, $\|M_2\| \leqslant \sqrt{m/n}\,O(\log^{3/2}(n))$.

Proof. We will again use the trace method and the first few manipulations are same as in the proof of the claim above. As in the proof above, we obtain the following:

$$\mathbb{E}[\mathrm{tr}((M_2M_2^\top)^\ell] = \sum_{i_1,\alpha_1,i_2,\alpha_2,\ldots,i_{2\ell},\alpha_{2\ell}} \mathbb{E}[\Pi_{j=1}^{2\ell} G(i_j,i_{j+1})]\,\mathbb{E}[\Pi_{j=1}^{2\ell} C_{i_j,i_{j+1}}(\alpha_j,\alpha_{j+1})].$$

We now analyze the above sum term by term. Observe that any entry of $C_{i,j}$ has mean-zero. Thus, if a term contributes non-zero value to the expectation, then the second product must have more than 1 entry for each edge $\{i_j, i_{j+1}\}$ of G. Thus, the number of distinct edges of G that appear in the first product in a term above is at most 2ℓ. As above, this implies that the number of distinct elements in $\{i_1, i_2, \ldots, i_{2\ell}\}$ for a term that contributes non-zero value to the expectation above is at most $\ell+1$. Let's call every collection of indices $(i_1, i_2, \ldots, i_{2\ell})$ that participates in some term with non-zero expectation, a "contributing" index.

Then, we can write the expression above as:

$$\mathbb{E}[\mathrm{tr}((M_2M_2^\top)^\ell] = \sum_{(i_1,i_2,\ldots,i_{2\ell}) \text{ contributing}} \mathbb{E}[\Pi_{j=1}^{2\ell} G(i_j,i_{j+1})] \sum_{\alpha_1,\alpha_2,\ldots,\alpha_{2\ell}} \mathbb{E}[\Pi_{j=1}^{2\ell} C_{i_j,i_{j+1}}(\alpha_j,\alpha_{j+1})]. \quad (5.10)$$

Next, by a Chernoff+Union bound argument, with probability at least $1-1/n$, for every i, j that is an edge in G, the number of non-zero entries in $C_{i,j}$ is at most $O(\sqrt{\log(n)}) \cdot 2^b$. Let's condition on this event in what follows.

Observe that the probability of any $q \leqslant 2\ell$ different entries of $C_{i,j}$ being simultaneously non-zero, conditioned on the event above is at most $O(\log^{q/2}(n)) \cdot 2^{-qb}$).

Next, fix the number of distinct α_js in the second sum above to γ. Then, observe that number of distinct pairs (α_j, α_{j+1}) must at least be $\gamma-1$ (start from α_1 and observe that each time a new vertex is encountered, there's also a new edge that is traversed. Thus, the number of edges is at least 1 less than the number of distinct vertices.). From the argument above, thus, the summation over α_js makes a total contribution of at most $\sum_{\gamma \leqslant 2\ell} 2^{b\gamma} O(\log^{(\gamma-1)/2}(n)) \cdot 2^{-(\gamma-1)b}) \leqslant 2\ell O(\log^\ell(n)) 2^b$.

Plugging this bound back in (5.10) then yields an upper estimate of $(\sum_{t \leqslant \ell+1} n^t (2m/n^2)^t) 2\ell O(\log^\ell(n)) 2^b$. Taking 2ℓth roots, we thus obtain that: $\mathbb{E}[\mathrm{tr}((M_2M_2^\top)^\ell]^{1/2\ell} \leqslant O(\log^{1/2}(n)) 2^{b/2\ell} \cdot O(m/n)^{1/2+1/2\ell}$. Using $\ell = \Omega(\log(n))$ then yields an upper bound of $\sqrt{m/n} O(\log^{3/2}(n))$ on the largest singular value of M_2 as required.

□

□

6 Lower Bound for Refuting Two-Block-Local PRGs

In this section, we establish that if $b > 10 \log\log(n)$, then there's no $2^{O(n/2^{4b})}$-time algorithm for image refutation of block-local PRG of stretch $\Omega(n2^b)$ based on the sum-of-squares method.

The main goal of this section is summarized in the following theorem.

Theorem 6.1. *For any $b > 10\log\log(n)$, there's a construction $\mathcal{G}\colon \{\pm1\}^n \to \{\pm1\}^m$ for $m = \Omega(n2^b)$ such that for any $z \in \{\pm1\}^m$, there's a feasible solution for the degree $\Theta(n/2^{4b})$ sum-of-squares relaxation of the constraints $\{\mathcal{G}_i = z_i\}$. In particular, sum of squares algorithm of degree $\Theta(n/2^{4b})$ cannot accomplish image refutation for $\mathcal{G}$.*

Our construction is extremely natural – the underlying graph will be chosen at random and the predicates (each edge has a different predicate) XORs on subsets of the blocks on the end points of an edge chosen with some care. We abstract our the properties of the random graph that we need by defining "nice" graphs below. We show that a random graph with appropriately chosen parameter immediately after.

6.1 The Constraint Graph

Definition 6.2 (Nice Graphs)**.** A (α, β)-nice graph G is a graph on $[n]$ such that:

1. G has $m = \Omega(n2^b)$ edges.
2. Degree of every vertex is at most 2^b.
3. Every induced subgraph of at most βn vertices has at most αn edges.

As we show next, (α, β)-nice graphs (for a proper choice of α and β) can be constructed by choosing G at random from the Erdős-Rényi distribution.

Lemma 6.3. *Fix $b \geqslant 10 \log\log(n)$. Let $G \sim G(n, q)$ for $q = C/n$ for $C = 2^b/10$. Then, for $\alpha = 1.4$ and $\beta = 2^{-4b}$, $\mathbb{P}[G$ is (α, β)-nice$] \geqslant 1 - o(1)$.*

Proof. Follows from Lemmas 6.4 and 6.5 below. □

Lemma 6.4. *Let $b \geqslant 10 \log\log(n)$. With probability at least $1 - 1/n$, a graph drawn according to $G(n, q)$ for $q = C/n$ for $C = 2^b/10$ has the maximum degree of its vertices upper bounded by $2^b - 1$.*

Proof. We assume that $2^b < n - 1$ as otherwise the claim is vacuous. The probability that the degree of a fixed vertex exceeds $2^b - 1$ is at most $\binom{n-1}{2^b}(C/n)^{2^b-1} \leqslant (eC/2^b)^{2^b-1} \leqslant 2^{-2^b}$. If $b \geqslant 10 \log\log(n)$, we are immediately done by a union bound over all possible n vertices. □

Lemma 6.5 (Vertex Expansion in Random Graphs)**.** *Fix $b \geqslant 10 \log\log(n)$. Let $G \sim G(n, q)$ for $q = C/n$ for $C = 2^b/10$. For $\beta = 2^{-4b}$ and $\alpha = 1.4$, with probability at least $1 - o(1)$ over the choice of G, every induced subgraph on at most $t < \beta n$ vertices has at most αt edges.*

Proof. Fix any t. By union bound, the probability that there's a t-size subgraph with more than αt edges is at most $\binom{n}{t} \cdot \binom{\binom{t}{2}}{\alpha t}(C/n)^{\alpha t}$. By standard approximations, this is at most $(ne/t)^t \cdot (te/\alpha)^{\alpha t} \cdot (C/n)^{\alpha t} \leqslant (t/n)^{t(\alpha-1)}(2C)^{\alpha t} \leqslant ((2C)^\alpha \cdot \beta^{\alpha-1}))^t$.

By a union bound, the probability that there's a t sized induced subgraph of G for $t < \beta n$ that has αt edges is at most: $2(2C)^\alpha \cdot \beta^{\alpha-1}$ which for the choice of $\beta = 2^{-4b}$ and $C < 2^b/10$ can be seen as at most $o(1)$. □

6.2 Lower Bound Instance

1. Let G be an (α, β)-nice graph.
2. We consider the seed $x \in \{\pm 1\}^{bn}$ as being divided into n blocks of b bits each as usual.
3. Fix i, a vertex of G. For every neighbor j of i, assign a distinct non-empty subset $S_{i,j} \subseteq \{(i, a) \mid a \in [b]\}$ to the edge $\{i, j\}$. Note that $S_{i,j}$ is distinct from $S_{j,i}$. Lemma 6.4 below shows that this is possible with high probability over the draw of the graph G in the first step.

4. Let $P_{i,j}(x_i, x_j) = \Pi_{v \in S_{i,j}, w \in S_{j,i}} x_v x_w$.

For large enough n and the instance chosen above, when $z \in \{0,1\}^m$ chosen uniformly at random, there's no solution $x \in \{\pm 1\}^b$ that satisfies more than $1/2 + \varepsilon$ fraction of the constraints by standard Chernoff+Union bound arguments. Nevertheless, we will construct a sum-of-squares solution - a pseudo-expectation - of degree $n/2^{4b}$ that satisfies all the XOR constraints for an arbitrary choice of z. This establishes that for $m = \Omega(n2^b)$, sum-of-squares of degree $n/2^{4b}$ cannot solve the image refutation problem for the PRG constructed in the previous section.

6.3 Constructing the Pseudo-expectation

We fix the following for this section.

1. We will fix G to be an (α, β)-nice graph on n vertices.
2. For every edge $\{i, j\}$ in G, let $S_{i,j}$s be the subsets of variables in the i^{th} block of b bits that participates in the XOR constraint for the edge $\{i, j\}$ chosen as in the previous section.
3. Order edges of G in any order and let $C_1, C_2, \ldots, C_m$ be defined by setting $C_e = S_{i,j} \cup S_{j,i}$ for $\{i, j\}$ being the eth edge. Observe that each C_i satisfies $|C_i| \leqslant 2b$.

Notation.

1. For any subsets Q, Q', $Q \Delta Q'$ denotes the symmetric difference of Q and Q'. More generally, for any subsets $Q_1, Q_2, \ldots, Q_r \subseteq [nb]$, we write $\Delta_{\ell \leqslant r} Q_i$ for the subset of all elements of $[nb]$ that occur in an odd number of sets in $Q_1, Q_2, \ldots, Q_r$.
2. For any collection of indices of constraints $Q \subseteq [m]$, we write $\mathcal{V}(Q)$ for the vertices of G that participate in some constraint in C_ℓ for some $\ell \in Q$.

A key notion that we will use in the construction of our pseudo-expectation is that of a low-degree derivation. Informally, low-degree derivation is a process that takes some XOR constraints and obtains new XOR constraints by combining already generated constraints with a certain restriction: two existing XOR constraints can be combined to "derive" a new constraint only if the symmetric difference of the sets of indices that the XOR constraints are on, is at most d - the degree of the derivation.

Definition 6.6 (Degree d Derivation)**.** A degree derivation of a subset $C \subseteq [nb]$ is a sequence of $T_1, T_2, \ldots, T_r$ of subsets of equations $C_1, C_2, \ldots, C_m$ in an XOR instance such that:

1. $\Delta_{\ell \in T_r} C_\ell = C$.
2. Each T_i is either C_j for some $j \leqslant m$ or
3. there exist $a, b < i$ such that $T_i = T_a \cup T_b$ and for $C_a = \Delta_{\ell \in T_a} C_\ell$ and $C_b = \Delta_{\ell \in T_b} C_\ell$, $|C_a \Delta C_b| \leqslant d$.

The key reason for sum-of-squares to not be able to solve the image refutation problem for the instance we constructed is that low-degree derivations cannot discover that the equations $x_{C_i} = z_i$ cannot be satisfied (for say, a randomly chosen z).

The following is a direct consequence of Lemma 6.5.

Lemma 6.7. *Let T be a collection of indices in $[m]$ such that $|\mathcal{V}(T)| < \beta n$. Then, $|\Delta_{\ell \in T} C_\ell| > 0.05|\mathcal{V}(T)|$.*

Proof. Let $Q = \mathcal{V}(T)$. Let E_Q be the edges corresponding to the constraints $\{C_\ell \mid \ell \in T\}$. Consider the vertices that have degree 1 or 2 in the graph (Q, E_Q). We will show that the number of such vertices is at least $0.1|Q|$. This is enough to complete the claim - this is because, for any block of variables corresponding to vertices of degree 1 or 2 in (Q, E_Q), the subset of variables involved in the constraints on the edges in E_Q are different and thus, there is at least one variable from the block that does occurs exactly once for every vertex of degree 1 or 2 in Q.

We now show that the degree 1 or 2 vertices in (Q, E_Q) are at least $0.1|\mathcal{V}(T)|$. Add any edge in G that is incident between vertices of Q and not already included in E_Q to obtain E'_Q. Observe that the degree 1 or 2 vertices in the graph (Q, E'_Q) are a subset of the degree 1 or 2 vertices in the graph (Q, E_Q). We will lower bound the degree 1 or 2 vertices in (Q, E'_Q) instead. Notice that this is the induced subgraph of G on vertices in Q. Further, $|Q| \leqslant \beta n$. Thus, applying Lemma 6.5, the number of edges $|E'_Q| \leqslant \alpha|Q| = 1.4|Q|$. Now, clearly, $\sum_{q \in Q} d_q = 2|E'_Q| \leqslant 2.8|Q|$, where d_q is the degree of a vertex $q \in Q$ in the subgraph (Q, E'_Q). In particular, $\sum_{q \in Q, d_q \geqslant 3} d_q \leqslant 2.8|Q|$ and thus, the number of vertices with degree at least 3 is at most $(2.8/3)|Q|$. Or, the number of vertices in (Q, E'_Q) of degree 1 or 2 is at least $0.2/3|Q| > 0.06|Q|$. □

Lemma 6.8 (Low-degree Derivations are Small)**.** *Let $d > 2b$ and suppose $100d \leqslant \beta n$. Then, if $T_1, T_2, \ldots, T_r$ is a degree d-derivation, then each $|\mathcal{V}(T_i)| \leqslant 50d$.*

Proof. Suppose, towards a contradiction, that there is a i such that $|\mathcal{V}(T_i)| > 50d$ and take the least such i. Then, $T_i = T_a \cup T_b$ for some $a, b < i$ such that $|\Delta_{\ell in T_i} C_\ell| = |\Delta_{\ell \in T_a \cup T_b} C_\ell| \leqslant d$. Thus, $|\mathcal{V}(T_a)|, |\mathcal{V}(T_b)| \leqslant 50d$. In particular, $|\mathcal{V}(T_i)| \leqslant 100d < \beta n$. Applying Lemma 6.7 to T_i, we have that $|\Delta_{\ell in T_i} C_\ell| > 0.05|\mathcal{V}(T_i)| \geqslant 0.05 * 50d = 2.5d$. This is contradiction as this means that $|\Delta_{\ell \in T_a \cup T_b} C_\ell| > d$. □

Corollary 6.9 (No low-degree derivations of $\emptyset$)**.** *Let $100d < \beta n$. Then, there's no degree d-derivation of $\emptyset$ in the XOR instance.*

Proof. Lemma 6.8 implies that set T of clauses produced by any degree d derivation satisfies $|\mathcal{V}(T)| \leqslant 50d$. Lemma 6.7 implies that $|\Delta_{\ell \in T} C_\ell| \neq 0$. Thus, there's no degree d derivation of $\emptyset$. □

Definition 6.10 (The Pseudo-Expectation)**.** Given an instance of XOR with equations $x_{C_i} = z_i$ for $1 \leqslant i \leqslant m$ and for any $z \in \{\pm 1\}^m$, define a linear operator $\tilde{\mathbb{E}}$ on mulitlinear monomials of degree $\leqslant d/4$ (and thus, all multilinear degree $\leqslant d$ polynomials)

1. $\tilde{\mathbb{E}}[x_\emptyset] = \tilde{\mathbb{E}}[1] = 1$.
2. $\tilde{\mathbb{E}}[x_{C_i}] = z_i$ for every $i \leqslant m$.
3. Repeat until impossible: if $\tilde{\mathbb{E}}[x_C], \tilde{\mathbb{E}}[x_{C'}]$ have been already set before, and $|C \Delta C'| \leqslant d/2$, then set $\tilde{\mathbb{E}}[x_{C \Delta C'}] = \tilde{\mathbb{E}}[x_C]\, \tilde{\mathbb{E}}[X_{C'}]$. Observe that this is equivalent to saying: If C has a degree $d/2$ derivation $T_1, T_2, \ldots, T_r$ of C from the instance, set $\tilde{\mathbb{E}}[x_C] = \Pi_{\ell \in T_r} z_\ell$.
4. Otherwise, set $\tilde{\mathbb{E}}[x_C] = 0$.

Key to the success of the procedure above for constructing the pseudo-expectation is the fact that there are no spurious low-degree derivations. This is the content of the following lemma.

Lemma 6.11 (Pseudo-expectation is well-defined)**.** *There's no C such that $\tilde{\mathbb{E}}[x_C]$ gets two conflicting values in Definition 6.10.*

Proof. If C has no degree $d/2$ derivation, there's nothing to prove. Suppose there's C such that $|C| \leqslant d/4$ and there are two distinct degree $d/2$ derivations of C. Then, combining the two derivations gives us a degree d derivation of $\emptyset$, and as a result, Corollary 6.9 yields a contradiction. □

Lemma 6.12 (Pseudo-expectation is PSD)**.** *For every degree $d/8$ polynomial p, $\tilde{\mathbb{E}}[p^2] \geqslant 0$.*

Proof. This argument is entirely analogous to the one in the classical Grigoriev's theorem (see lecture notes [BS17]). We define an equivalence relation on monomials of degree at most $d/8$. Two monomial indices $C \sim C'$ (are in the same equivalence class) if $\tilde{\mathbb{E}}[x_{C\Delta C'}] \neq 0$. We claim that this is indeed an equivalence relation. It is easy to see that $C \sim C$ (reflexivity). To check transitivity, suppose $C_1 \sim C_2$ and $C_2 \sim C_3$. Then, $C_1\Delta C_2$ has a degree $d/2$ derivation and $C_2\Delta C_3$ has a degree $d/2$ derivation. Further, $|C_1\Delta C_2|, |C_2\Delta C_3| \leqslant d/4$ since each C_i has size at most $d/8$. Thus, $|C_1\Delta C_3| \leqslant |C_1\Delta C_2| + |C_2\Delta C_3| \leqslant d/2$ and thus, $C_1\Delta C_3$ has a degree $d/2$-derivation and in particular, $\tilde{\mathbb{E}}[x_{C_1\Delta C_3}] \neq 0$ implying that $C_1 \sim C_3$.

Let $p_1, p_2, \ldots p_r$ be such that $p = \sum_i p_i$ and p_i contains only the monomials from p that belong to the ith equivalence class. Then, observe that $\tilde{\mathbb{E}}[p_i^2] = \sum_{C,C'} p_i(C)p_i(C')\,\tilde{\mathbb{E}}[x_{C\Delta C'}] = \sum_{C,C'} p_i(C)p_i(C')\,\tilde{\mathbb{E}}[x_C]\,\tilde{\mathbb{E}}[x_{C'}] = (\sum_C p_i(C)\,\tilde{\mathbb{E}}[x_C])^2 \geqslant 0$. To finish the proof, observe that $\tilde{\mathbb{E}}[p^2] = \sum_{i,j} \tilde{\mathbb{E}}[p_ip_j] = \sum_i \tilde{\mathbb{E}}[p_i^2] \geqslant 0$ where we used the fact that $\tilde{\mathbb{E}}[p_ip_j] = 0$ since the pseudo-expectation of product of monomials from different equivalence classes is 0.

□

Our main result about this construction is the following claim that shows that the sum-of-squares relaxation's value for the maximum of the polynomial $\sum_{\{i,j\}\in G} P_{i,j}(x)$ equals m.

Lemma 6.13 (Formal version of Theorem 6.1)**.** *For the instance considered in this section and any $z \in \{\pm 1\}^m$, there exists a pseudo-expectation $\tilde{\mathbb{E}}$ of degree $\Omega(n/2^{4b})$ that satisfies the set of constraints $\{p_{i,j}(x) = z_{i,j}\}$. In particular, $\tilde{\mathbb{E}}[\sum_{\{i,j\}\in G} z_{i,j}p_{i,j}(x)] = m$.*

Proof. We use the construction described above. Lemma 6.11 shows that the construction is well-defined. By our choice, $\tilde{\mathbb{E}}$ is defined only on multi-linear monomials and extended to other monomials by standard multilinear reduction. Thus, it satisfies the constraints that $x_{i,j}^2 = 1$. Further, for any monomial x_C with C of size at most $d/4 - |C_i|$, $\tilde{\mathbb{E}}[x_Cx_{C_i}] = \tilde{\mathbb{E}}[x_C]z_i$. This is because either $\tilde{\mathbb{E}}[x_Cx_{C_i}] = 0$ in which case the claim is vacuous or otherwise, C, C_i are obtained in degree d derivation in which case they are set to satisfy the constraint above. Finally, Lemma 6.12 shows that $\tilde{\mathbb{E}}[p^2] \geqslant 0$. This completes the proof. □

7 A class of block-local candidate pseudorandom generators

In this section we outline a simple candidate pseudorandom generator of degree d that has potentially output length as large as $n^{d/2-\varepsilon}$. We have not conducted an extensive study of this candidate's security, but do believe it's worthwhile example as a potential counterpoint to our results on limitations for pseudorandom generator, demonstrating that they might be tight.

The idea is simple: for a finite group $\mathbb{G}$ that does not have any abelian quotient group (for example, a non-abelian simple group will do), we choose dm random indices $\{i_{j,k}\}_{j\in[m],k\in[d]}$ and let

$\mathcal{G}$ be the generator mapping $\mathbb{G}^n$ to $\mathbb{G}^m$ where

$$\mathcal{G}(x)_j = x_{i_{j,1}} * x_{i_{j,2}} * \cdots * x_{i_{j,d}} \tag{7.1}$$

If want to output m bits rather than m elements of $\mathbb{G}$, then we use a group $\mathbb{G}$ of even order and apply to each coordinate some balanced map $f : \mathbb{G} \to \{0,1\}$. For every group element $g \in \mathbb{G}$, the predicate

$$x_1 * \cdots * x_d = g \tag{7.2}$$

supports a $d-1$ wise independent distribution. Hence, using the results of [KMOW17] we can show that as long $m < n^{d/2-\varepsilon}$, for a random $z \in \mathbb{G}^m$, the SOS algorithm cannot be used to efficiently refute the statement that $z = \mathcal{G}(x)$ for some x.

Ruling out Gaussian-elimination type attacks is trickier. For starters, solving a linear system over a non-abelian group is NP-hard [GR02, KTT07]. Also, Applebaum and Lovett [AL16, Theorem 5.5] showed that at least for the large d case, because the predicate (7.2) has rational degree d, the image-refutation problem for this generator is hard with respect to algebraic attacks (that include Gaussian elimination) for $m = n^{\Omega(d)}$. Nevertheless, there are non trivial algorithms in the group theoretic settings (such as the low index subgroup algorithm, see [CD05] and [RSW06, Sec. 6]). A more extensive study of algebraic attacks against this predicate is needed to get better justifications of its security, and we leave such study for future work.

We remark that the condition that the group $\mathbb{G}$ does not have abelian normal subgroups is crucial. Otherwise, we can write $\mathbb{G}$ as the direct product $\mathbb{H} \times \mathbb{H}'$ where $\mathbb{H}$ is abelian, and project all equations to their component in $\mathbb{H}$. We will get m random equations in n variables over the abelian group $\mathbb{H}$, and hence we can use Gaussian elimination to refute those.

References

[ABKS17] Prabhanjan Ananth, Zvika Brakerski, Dakshita Khurana, and Amit Sahai, *Private communication*, 2017. 1, 9

[ABW10] Benny Applebaum, Boaz Barak, and Avi Wigderson, *Public-key cryptography from different assumptions*, Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC, ACM, 2010, pp. 171–180. 1

[AIK06] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz, *Cryptography in NC^0*, SIAM J. Comput. **36** (2006), no. 4, 845–888. 1, 5, 40, 42

[AJ15] Prabhanjan Ananth and Abhishek Jain, *Indistinguishability obfuscation from compact functional encryption*, Advances in Cryptology - CRYPTO, 2015, pp. 308–326. 42

[AJS15] Prabhanjan Ananth, Abhishek Jain, and Amit Sahai, *Indistinguishability obfuscation from functional encryption for simple functions*, IACR Cryptology ePrint Archive **2015** (2015), 730. 1, 42

[AL16] Benny Applebaum and Shachar Lovett, *Algebraic attacks against random local functions and their countermeasures*, STOC, ACM, 2016, pp. 1087–1100. 34

[AOW15a] Sarah R. Allen, Ryan O'Donnell, and David Witmer, *How to refute a random CSP*, FOCS, IEEE Computer Society, 2015, pp. 689–708. 5, 6, 10, 13

[AOW15b] Sarah R. Allen, Ryan O'Donnell, and David Witmer, *How to refute a random CSP*, 2015 IEEE 56th Annual Symposium on Foundations of Computer Science—FOCS 2015, IEEE Computer Soc., Los Alamitos, CA, 2015, pp. 689–708. MR 3473335 13

[App13] Benny Applebaum, *Pseudorandom generators with long stretch and low locality from random local one-way functions*, SIAM J. Comput. **42** (2013), no. 5, 2008–2037. 1, 5

[App16] _____, *Cryptographic hardness of random local functions - survey*, Computational Complexity **25** (2016), no. 3, 667–722. 1

[AR16] Benny Applebaum and Pavel Raykov, *Fast pseudorandom functions based on expander graphs*, Theory of Cryptography - 14th International Conference, TCC 2016-B, vol. 9985, 2016, pp. 27–56. 1, 5

[AS16] Prabhanjan Ananth and Amit Sahai, *Projective arithmetic functional encryption and indistinguishability obfuscation from degree-5 multilinear maps*, IACR Cryptology ePrint Archive **2016** (2016), 1097. 1, 8

[BCK15] Boaz Barak, Siu On Chan, and Pravesh K. Kothari, *Sum of squares lower bounds from pairwise independence [extended abstract]*, STOC'15—Proceedings of the 2015 ACM Symposium on Theory of Computing, ACM, New York, 2015, pp. 97–106. MR 3388187 6

[BF01] Dan Boneh and Matthew K. Franklin, *Identity-based encryption from the weil pairing*, in Kilian [Kil01], pp. 213–229. 8

[BGH+15] Zvika Brakerski, Craig Gentry, Shai Halevi, Tancrède Lepoint, Amit Sahai, and Mehdi Tibouchi, *Cryptanalysis of the quadratic zero-testing of GGH*, Cryptology ePrint Archive, Report 2015/845, 2015. 8

[BGI+01] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang, *On the (im)possibility of obfuscating programs*, in Kilian [Kil01], Full version in [BGI+12], pp. 1–18. 7

[BGI+12] _____, *On the (im)possibility of obfuscating programs*, J. ACM **59** (2012), no. 2, 6:1–6:48. 35

[BRS11] Boaz Barak, Prasad Raghavendra, and David Steurer, *Rounding semidefinite programming hierarchies via global correlation*, 2011 IEEE 52nd Annual Symposium on Foundations of Computer Science—FOCS 2011, IEEE Computer Soc., Los Alamitos, CA, 2011, pp. 472–481. MR 2932723 6

[BS17] Boaz Barak and David Steurer, *Proofs, beliefs, and algorithms through the lens of sum-of-squares*, 2017, Lecture notes, available on `http://sumofsquares.org`. 4, 18, 33, 39

[BV15] Nir Bitansky and Vinod Vaikuntanathan, *Indistinguishability obfuscation from functional encryption*, IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS, IEEE Computer Society, 2015, pp. 171–190. 42

[CD05] Marston Conder and Peter Dobcsányi, *Applications and adaptations of the low index subgroups procedure*, Mathematics of computation **74** (2005), no. 249, 485–497. 34

[CFL+16] Jung Hee Cheon, Pierre-Alain Fouque, Changmin Lee, Brice Minaud, and Hansol Ryu, *Cryptanalysis of the new CLT multilinear map over the integers*, Cryptology ePrint Archive, Report 2016/135, 2016. 8

[CGH+15] Jean-Sébastien Coron, Craig Gentry, Shai Halevi, Tancrède Lepoint, Hemanta K. Maji, Eric Miles, Mariana Raykova, Amit Sahai, and Mehdi Tibouchi, *Zeroizing without low-level zeroes: New MMAP attacks and their limitations*, Advances in Cryptology – CRYPTO '15, 2015, pp. 247–266. 8

[CHL+15] Jung Hee Cheon, Kyoohyung Han, Changmin Lee, Hansol Ryu, and Damien Stehlé, *Cryptanalysis of the multilinear map over the integers*, Advances in Cryptology – EUROCRYPT '15, 2015, pp. 3–12. 8

[CJL16] Jung Hee Cheon, Jinhyuck Jeong, and Changmin Lee, *An algorithm for NTRU problems and cryptanalysis of the GGH multilinear map without an encoding of zero*, Cryptology ePrint Archive, Report 2016/139, 2016. 8

[CLR15] Jung Hee Cheon, Changmin Lee, and Hansol Ryu, *Cryptanalysis of the new CLT multilinear maps*, Cryptology ePrint Archive, Report 2015/934, 2015. 8

[CLT13] Jean-Sébastien Coron, Tancrède Lepoint, and Mehdi Tibouchi, *Practical multilinear maps over the integers*, Advances in Cryptology - CRYPTO, 2013, pp. 476–493. 8

[CLT15] _____, *New multilinear maps over the integers*, Advances in Cryptology - CRYPTO, 2015, pp. 267–286. 8

[CM01] Mary Cryan and Peter Bro Miltersen, *On pseudorandom generators in NC*, 26th International Symposium on Mathematical Foundations of Computer Science, MFCS, 2001, pp. 272–284. 5

[CW04] Moses Charikar and Anthony Wirth, *Maximizing quadratic programs: Extending grothendieck's inequality*, FOCS, IEEE Computer Society, 2004, pp. 54–60. 10, 15, 17, 40

[Fei02] Uriel Feige, *Relations between average case complexity and approximation complexity*, Proceedings of the Thirty-Fourth Annual ACM Symposium on Theory of Computing, ACM, New York, 2002, pp. 534–543 (electronic). MR 2121179 5

[Fei07] Uriel Feige, *Refuting smoothed 3CNF formulas*, FOCS, IEEE Computer Society, 2007, pp. 407–417. 5

[FO07] Uriel Feige and Eran Ofek, *Easily refutable subformulas of large random 3CNF formulas*, Theory Comput. **3** (2007), 25–43. MR 2322854 5

[GGH13a] Sanjam Garg, Craig Gentry, and Shai Halevi, *Candidate multilinear maps from ideal lattices*, Advances in Cryptology - EUROCRYPT, 2013, pp. 1–17. 8

[GGH+13b] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters, *Candidate indistinguishability obfuscation and functional encryption for all circuits*, 54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 2013, pp. 40–49. 7

[GGH15] Craig Gentry, Sergey Gorbunov, and Shai Halevi, *Graph-induced multilinear maps from lattices*, Theory of Cryptography - 12th Theory of Cryptography Conference, TCC, 2015, pp. 498–527. 8

[GLS81] M. Grötschel, L. Lovász, and A. Schrijver, *The ellipsoid method and its consequences in combinatorial optimization*, Combinatorica **1** (1981), no. 2, 169–197. MR 625550 39

[Gol00] Oded Goldreich, *Candidate one-way functions based on expander graphs*, Electronic Colloquium on Computational Complexity (ECCC) **7** (2000), no. 90. 1, 5

[GR02] Mikael Goldmann and Alexander Russell, *The complexity of solving equations over finite groups*, Inf. Comput. **178** (2002), no. 1, 253–262. 5, 34

[Had00] Satoshi Hada, *Zero-knowledge and code obfuscation*, Advances in Cryptology - ASIACRYPT 2000, 6th International Conference on the Theory and Application of Cryptology and Information Security (Tatsuaki Okamoto, ed.), Lecture Notes in Computer Science, vol. 1976, Springer, 2000, pp. 443–457. 7

[HJ15] Yupu Hu and Huiwen Jia, *Cryptanalysis of GGH map*, Cryptology ePrint Archive, Report 2015/301, 2015. 8

[IK02] Yuval Ishai and Eyal Kushilevitz, *Perfect constant-round secure computation via perfect randomizing polynomials*, Automata, Languages and Programming, 29th International Colloquium, ICALP, 2002, pp. 244–256. 40, 42

[IKO+11] Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, Manoj Prabhakaran, and Amit Sahai, *Efficient non-interactive secure computation*, Advances in Cryptology - EUROCRYPT, 2011, pp. 406–425. 1

[Jou00] Antoine Joux, *A one round protocol for tripartite diffie-hellman*, Algorithmic Number Theory, 4th International Symposium, ANTS-IV, Leiden, The Netherlands, July 2-7, 2000, Proceedings (Wieb Bosma, ed.), Lecture Notes in Computer Science, vol. 1838, Springer, 2000, pp. 385–394. 8

[Kil01] Joe Kilian (ed.), *Advances in cryptology - crypto 2001, 21st annual international cryptology conference, santa barbara, california, usa, august 19-23, 2001, proceedings*, Lecture Notes in Computer Science, vol. 2139, Springer, 2001. 35

[KMOW17] Pravesh K. Kothari, Ryuhei Mori, Ryan O'Donnell, and David Witmer, *Sum of squares lower bounds for refuting any CSP*, CoRR **abs/1701.04521** (2017). 4, 5, 6, 34

[KTT07] Ondrej Klíma, Pascal Tesson, and Denis Thérien, *Dichotomies in the complexity of solving systems of equations over finite semigroups*, Theory of Computing Systems **40** (2007), no. 3, 263–297. 34

[Las01] Jean B. Lasserre, *New positive semidefinite relaxations for nonconvex quadratic programs*, Advances in convex analysis and global optimization (Pythagorion, 2000), Nonconvex Optim. Appl., vol. 54, Kluwer Acad. Publ., Dordrecht, 2001, pp. 319–331. MR 1846160 4, 39

[Lin16a] Huijia Lin, *Indistinguishability obfuscation from constant-degree graded encoding schemes*, Advances in Cryptology - EUROCRYPT, 2016, pp. 28–57. 1, 8, 40

[Lin16b] _____, *Indistinguishability obfuscation from DDH on 5-linear maps and locality-5 PRGs*, IACR Cryptology ePrint Archive (2016), 1096. 1, 8, 40, 41, 42, 43

[LMSS07] Nati Linial, Shahar Mendelson, Gideon Schechtman, and Adi Shraibman, *Complexity measures of sign matrices*, Combinatorica **27** (2007), no. 4, 439–463. MR 2359826 24

[LS09] Nati Linial and Adi Shraibman, *Learning complexity vs. communication complexity*, Combin. Probab. Comput. **18** (2009), no. 1-2, 227–245. MR 2497381 24

[LT17] Huijia Lin and Stefano Tessaro, *Indistinguishability obfuscation from bilinear maps and block-wise local prgs*, IACR Cryptology ePrint Archive (2017), 250. 1, 3, 6, 7, 8, 9, 12, 18, 19, 25, 40, 41, 42

[LV16] Huijia Lin and Vinod Vaikuntanathan, *Indistinguishability obfuscation from ddh-like assumptions on constant-degree graded encodings*, IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS, IEEE Computer Society, 2016, pp. 11–20. 1, 8, 41, 42

[LV17a] Alex Lombardi and Vinod Vaikuntanathan, *Minimizing the complexity of goldreich's pseudorandom generator*, IACR Cryptology ePrint Archive (2017), 277. 2, 4

[LV17b] _____, *On the non-existence of blockwise 2-local prgs with applications to indistinguishability obfuscation*, IACR Cryptology ePrint Archive **2017** (2017), 301. 1, 2, 3, 4, 6, 7, 12, 13

[MF15] Brice Minaud and Pierre-Alain Fouque, *Cryptanalysis of the new multilinear map over the integers*, Cryptology ePrint Archive, Report 2015/941, 2015. 8

[MST06] Elchanan Mossel, Amir Shpilka, and Luca Trevisan, *On epsilon-biased generators in NC^0*, Random Struct. Algorithms **29** (2006), no. 1, 56–81. 1, 4, 5, 8

[MSZ16] Eric Miles, Amit Sahai, and Mark Zhandry, *Annihilation attacks for multilinear maps: Cryptanalysis of indistinguishability obfuscation over GGH13*, Cryptology ePrint Archive, Report 2016/147, 2016. 8

[O'D14] Ryan O'Donnell, *Analysis of boolean functions*, Cambridge University Press, 2014. 14

[OW14] Ryan O'Donnell and David Witmer, *Goldreich's PRG: evidence for near-optimal polynomial stretch*, IEEE 29th Conference on Computational Complexity—CCC 2014, IEEE Computer Soc., Los Alamitos, CA, 2014, pp. 1–12. MR 3280991 6

[Par00] Pablo A Parrilo, *Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization*, Ph.D. thesis, Citeseer, 2000. 4, 39

[RRS16] Prasad Raghavendra, Satish Rao, and Tselil Schramm, *Strongly refuting random csps below the spectral threshold*, CoRR **abs/1605.00058** (2016). 4, 5

[RSW06] Eyal Rozenman, Aner Shalev, and Avi Wigderson, *Iterative construction of cayley expander graphs.*, Theory OF Computing **2** (2006), no. 5, 91–120. 34

[Sho87] N. Z. Shor, *Quadratic optimization problems*, Izv. Akad. Nauk SSSR Tekhn. Kibernet. (1987), no. 1, 128–139, 222. MR 939596 4, 39

[SW14] Amit Sahai and Brent Waters, *How to use indistinguishability obfuscation: deniable encryption, and more*, Symposium on Theory of Computing, STOC, ACM, 2014, pp. 475–484. 8

[Wit17] David Witmer, *On refutation of random constraint satisfaction problems (thesis proposal)*, 2017, http://www.cs.cmu.edu/~dwitmer/papers/proposal.pdf. 3, 5

A Analysis of the basic SDP program

The degree d SOS program [BS17] for a polynomial optimization problem of the form

$$\max_{x\in\{\pm 1\}^n} p(x)$$

corresponds to

$$\max_{\mu} \tilde{\mathbb{E}}\, p$$

where $\tilde{\mathbb{E}}$ ranges over the set of degree d expectation operators that satisfy the constraints $\{x_i^2 = 1\}_{i=1}^n$. These are defined as follows:

Definition A.1 (Pseudo-expectation)**.** Let $\mathcal{P}_{n,d}$ denote the space of all degree $\leqslant d$ polynomials on n variables. A linear operator $\tilde{\mathbb{E}} : \mathcal{P}_{n,d}$ is a degree d *pseudo-expectation* if it satisfies the following conditions:

1. $\tilde{\mathbb{E}}[1] = 1$.
2. $\tilde{\mathbb{E}}[p^2] \geqslant 0$ for every polynomial p of degree at most $d/2$.

A pseudo-expectation is said to satisfy a constraint $\{q = 0\}$ if for every polynomial p of degree at most $d - deg(q)$, $\tilde{\mathbb{E}}[pq = 0$. We say that $\tilde{\mathbb{E}}$ satisfies the constraint $\{q \geqslant 0\}$ if for every polynomial p of degree at most $d/2 - deg(q)/2$, $\tilde{\mathbb{E}}[p^2 q] \geqslant 0$.

If μ is any distribution on $\mathbb{R}^n$, then the associated expectation is a pseudo-expectation operator of all degrees. The above definition can be thought of as a relaxation of the notion of an actual expectation.

Key to the utility of the definition above is the following theorem that shows one can efficiently search over the space of all degree d pseudo-expectations.

Theorem A.2 ([Sho87, Par00, Las01])**.** *For any n, and integer d, the following set has an $n^{O(d)}$ time weak separation oracle (in the sense of [GLS81]):*

$$\{\tilde{\mathbb{E}}[(1, x_1, x_2, \ldots, x_n,)^{\otimes d}] \mid \tilde{\mathbb{E}} \text{ is a degree } d \text{ pseudo-expectation}\}$$

In this appendix we expand on how Charikar and Wirth's work [CW04] implies the the following theorem:

Theorem A.3. *For every degree two polynomial $p : \mathbb{R}^n \to \mathbb{R}$ with no constant term, the value of the degree two SOS program for*

$$\max_{x \in \{\pm 1\}^n} p(x) \tag{A.1}$$

is larger than the true value of (A.1) by a factor of at most $O(\log n)$.

Theorem A.3 is a direct implication of the following result of [CW04]:

Theorem A.4 (Symmetric Grothendieck Inequality, [CW04], Theorem 1)**.** *Let A be any $m \times m$ matrix such that $A_{i,i} = 0$ for every i. Then,*

$$\max_{X \succeq 0, X_{i,i}=1 \forall i} Tr(AX) \leqslant O(\log n) \max_{x \in \{\pm 1\}^n} x^\top A x$$

Proof of Theorem A.3 from Theorem A.4. Suppose that there is a degree 2 pseudo-distribution $\{x\}$ such that $\tilde{\mathbb{E}}\, p(x) \geqslant \theta$, and let X be the $n+1 \times n+1$ matrix corresponding to $\tilde{\mathbb{E}}(x,1)(x,1)^\top$. That is, $X_{i,j} = \tilde{\mathbb{E}}\, x_i x_j$ and $X_{n+1,i} = X_{i,n+1} = \tilde{\mathbb{E}}\, x_i$. Note that X is a psd matrix with 1's on the diagonal.

Then $\mathrm{Tr}(AX) \geqslant \theta$ if A be the $(n+1) \times (n+1)$ matrix that represents the polynomial p. In this case Theorem A.4 implies that there is an $n+1$ dimensional vector $(x, \sigma) \in \{\pm 1\}^{n+1}$ such that $(x,\sigma)^\top A (x,\sigma) \geqslant \Omega(\theta/\log n)$. If we write $p(x) = q(x) + l(x)$, where q is the homogeneous degree two and l is linear, then we can see by direct inspection that

$$(x,\sigma)^\top A (x,\sigma) = q(x) + \sigma l(x) = p(\sigma x)$$

with the last equality following from the fact that $q(-x) = q(x)$ and $l(-x) = -l(x)$. Hence the vector $\sigma x \in \{\pm 1\}^n$ demonstrates that the value of (A.1) is at least $\Omega(\theta/\log n)$. □

B The Lin-Tessaro candidate obfuscator

In this section we provide more information on the candidate obfuscator of Lin and Tessaro [LT17] and its relation to the notion of block-wise local PRGs. At a very high level, the construction, which builds on a line of works initiated by the beautiful paper of Lin [Lin16a], can be described as follows:

- They reduce the task of building an obfuscator to the task of constructing a *functional encryption* scheme[15] for NC_1 functions that has a certain *ciphertext compactness* property.
- Lin [Lin16b] showed that one can obtain an appropriate functional encryption schemes for ℓ-degree functions from ℓ-linear maps.
- The idea behind closing the gap between NC_1 functions and degree ℓ maps is to use *randomized encodings* [IK02, AIK06] which encode an NC_1 function f of complexity m by a function g with *constant* input locality that takes an additional random input of length m.
- Unfortunately, encoding this random input would destroy the compactness property and hence we instead encode the *seed* for a pseudorandom generator $G : \{0,1\}^n \to \{0,1\}^m$.

[15] This is an encryption scheme that supports generation of restricted decryption keys that allow computation on encrypted values.

- If every output of the pseudorandom generator is a polynomial of degree d and every output of the randomized encoding depends on at most c bits of the input, we get that the resulting function is a polynomial of degree dc, which can then be evaluated using a dc-linear map.

- To reduce the need for multilinearity further, one can *pre-process* the seed for the generator. Thus, instead of encoding only the seed, we also encode the values of various monomials applied to it. To maintain the compactness property, the stretch of the pseudorandom generator needs to compensate for this preprocessing. If every output of the generator depends on s monomials which are arranged in a "nice" way (as happens to be the case of block locality), then this preprocessing will reduce the final degree to d but increase the length of the seed by a factor of s^c, where c is the locality of the randomized encoding scheme (i.e., the number of bits from the random tape that each output depends upon). Thus, loosely speaking, the output length of a pseudorandom generator of degree d and sparsity s will need to be roughly $s^c n^{1+\varepsilon}$ for it to be applicable in this setting. For known randomized encodings, the locality parameter c is at least 3, which in particular means that for block local transformations of block length b, we require an output length of at least $2^{3b} n^{1+\varepsilon}$. Our results show that such output length cannot be achieved with a two block-local generator.

We now provide a somewhat more detailed, though still quite informal, overview of the recent construction of obfuscations and how simple pseudorandom generators fit into this picture. See the introduction of [LT17] for a more complete description of the history and technical tools.

All current candidate constructions of indistinguishability obfuscation (iO) rely on the notion of multilinear maps (or a related notion called graded encoding schemes). Very roughly speaking, an ℓ-linear map allows one to evaluate any degree-ℓ polynomials on secret encoded values and to test whether the output of such a polynomial is zero or not.

Having an ℓ-linear map in hand, constructions of iO usually proceed in two steps. The first step is generic: the goal is to base the existence of iO on the seemingly weakest possible generic primitive. The second step is to design a construction of this latter primitive using an ℓ-linear map for ℓ being as small as possible (preferably, $\ell = 2$).

As it turns out the right notion that on the one hand is strong enough to allow for bootstrapping to iO but on the other is simple enough that we can construct it from ℓ-linear maps is *functional encryption* (FE). A functional encryption scheme supports (in addition to encryption and decryption) restricted decryption keys that allow users to learn specific functions of the encrypted data and nothing else. That is, the holder of the secret key of the scheme can generate a key for a function f and whoever holds a ciphertext of a message x and the key for f can compute $f(x)$ but gain no additional information about x.

Lin [Lin16b] showed how to construct an FE scheme that supports all degree-ℓ functions using an ℓ-linear map. Furthermore, Lin's construction has various efficiency properties that are useful when using it to get all the way to iO and in particular the crucial property of *ciphertext compactness* which roughly means that the ciphertext can be smaller than the description of the function. The security of the construction is reduced to an assumption called Symmetric External Diffie-Hellman (SXDH) on the ℓ-linear map. For $\ell = 2$ this assumption is quite common.

Having the second step of our construction, we proceed with the first step – getting a construction of iO for all circuits from FE for low degree polynomials. Before we explain the approach of Lin and Tessaro [LT17] it is useful to explain the approach of Lin and Vaikuntanathan [LV16]. The first

step was to construct FE for NC_1 with some compactness property[16] rather than plain iO for all circuits. This is enough by a bootstrapping theorem of [AJ15, BV15]. The next step was to notice (see [AJS15]) that it is actually enough to be able to construct an FE for NC_0 and assume a PRG in NC_0 with polynomial stretch. The idea of this transformation is to translate functions in NC_1 into their randomized encoding[17] which is in NC0 [IK02, AIK06]. A randomized encoding, as the name suggests, is a randomized procedure so we need to encode the randomness for evaluation. It turns out that the randomness size is proportional to the function size which, if we embed in a ciphertext, causes us to lose compactness altogether. Thus, to preserve compactness, the randomness is derived via a PRG. To preserve the low degree of the computation, we further need the PRG to be local. This resulted in [LV16] with a construction of iO from $O(1)$-linear maps using a constant locality PRGs.

The next step was made by Lin in [Lin16b], where she was able to optimize the above approach and obtain a construction with the concrete constant 5. Specifically, Lin proved a generic theorem that says that any locality-ℓ PRG can be used together with an FE scheme that supports polynomials of degree ℓ to get iO.[18] As we have already mentioned, in the above statement ℓ cannot be smaller than 5 as locality 4 PRG do not exist. The idea of Lin was to reduce the degree of the polynomial to be evaluated by *preprocessing* some of the computation of the function already at the time of encryption. To illustrate this idea, think of a function $f(x, y)$ that is linear in x but say quadratic in y. If we pre-compute $x \otimes y$ (where $\otimes$ denotes tensor product), then we can compute f with one degree less: there exists a function $f'(x, y, x \otimes y)$ that computes $f(x, y)$ in degree 2, by replacing each monomial of the form xy_1y_2 with a monomial of the form $(xy_1)y_2$ and taking (xy_1) from $x \otimes y$. Deciding what to pre-compute and what not is a delicate task as we have to keep the ciphertext compact (so we clearly cannot pre-compute all possible monomials of f).

In the work of Lin and Tessaro [LT17] they propose a variant of local PRG called block-wise local PRGs (see Section 5) to circumvent the $\ell \geqslant 5$ barrier and were able to show how to pre-process the inputs in the new construction correctly to preserve the various efficiency properties. Their theorem (roughly speaking) is that for any ℓ, iO can be constructed from an ℓ-local PRG with blocks of size logarithmic in the security parameter and polynomial stretch and ℓ-linear maps. Let us elaborate a little more on how the transformation works. We refer to [LT17] for the full details.

Recall that we have reduced the task of constructing an iO scheme to the task of building FE for NC_0 functionalities by using a randomized encodings. Given a function g in NC_1, we will generate a functional key for the randomized encoding of the function with randomness derived from a PRG.

$$\hat{g}(x, s) = \mathsf{RandEnc}(g, x; \mathsf{PRG}(s)).$$

This is a function in NC_0. Its degree depends on the degree of the PRG and the degree of the randomized encoding (viewed as polynomials over the rationals). It is known that the degree of the randomized encoding is 1 in x and 3 in $\mathsf{PRG}(s)$. So (very roughly) this can be re-written as

$$\hat{g}(x, s) = \sum_{i_0, i_1, i_2, i_3} c_{i_0, i_1, i_2, i_3} x_{i_0} r_{i_1} r_{i_2} r_{i_3},$$

[16] Compactness says, roughly, that the size of a ciphertext in the FE scheme is sub-linear in the size of the function for which we generate a key.

[17] Randomizing encodings allow to represent a function $f(x)$ by a low-degree randomized mapping $\hat{f}(x, r)$ whose output distribution on an input x is a randomized encoding of $f(x)$.

[18] Here and in the rest of this section we ignore additional assumptions (such as Learning With Errors or the need for sub-exponential hardness) that are sometimes required due to technical reasons we will not go into.

where $r = \mathsf{PRG}(s)$. This apparently is not enough for the efficient preprocessing, so [Lin16b] suggests a different way to compute the r_i's: she uses 3 seeds per random string and defines

$$r_{i_1} r_{i_2} r_{i_3} = \mathsf{PRG}(s_{i_1})\mathsf{PRG}(s_{i_2})\mathsf{PRG}(s_{i_3}).$$

The latter has a very small number of degree 3 monomials which allows her to pre-process them without compromising compactness of the scheme.

The case of block-wise local PRGs is handled in a very similar way. The monomials are written in the same way but now the preprocessing is over a much larger set. They first pre-process all possible symbols per block and then pre-compute all degree-3 monomials over theses symbols. For the scheme to stay compact the PRG thus has to map $2^b n$ bits into roughly $(2^b)^3 \cdot n^{1+\varepsilon}$ bits for some constant $\varepsilon > 0$.

ECCC ISSN 1433-8092
https://eccc.weizmann.ac.il