Rigid Matrices From Rectangular PCPs
or: Hard Claims Have Complex Proofs

Amey Bhangale*†, Prahladh Harsha‡†, Orr Paradise§† and Avishay Tal§
*Department of Computer Science and Engineering
University of California, Riverside, CA, USA
Email: ameyb@ucr.edu
‡School of Technology and Computer Science
Tata Institute of Fundamental Research, Mumbai, India
Email: prahladh@tifr.res.in
§Department of Electrical Engineering and Computer Sciences
University of California, Berkeley, CA, USA
Emails: orrp@cs.berkeley.edu, atal@berkeley.edu

***Abstract*—We introduce a variant of PCPs, that we refer to as *rectangular* PCPs, wherein proofs are thought of as square matrices, and the random coins used by the verifier can be partitioned into two disjoint sets, one determining the *row* of each query and the other determining the *column*.**

We construct PCPs that are *efficient*, *short*, *smooth* and (almost-)*rectangular*. As a key application, we show that proofs for hard languages in $\mathsf{NTIME}(2^n)$, when viewed as matrices, are rigid infinitely often. This strengthens and simplifies a recent result of Alman and Chen [FOCS, 2019] constructing explicit rigid matrices in FNP. Namely, we prove the following theorem:

- **There is a constant $\delta \in (0,1)$ such that there is an FNP-machine that, for infinitely many N, on input 1^N outputs $N \times N$ matrices with entries in $\mathbb{F}_2$ that are δN^2-far (in Hamming distance) from matrices of rank at most $2^{\log N/\Omega(\log\log N)}$.**

Our construction of rectangular PCPs starts with an analysis of how randomness yields queries in the Reed–Muller-based outer PCP of Ben-Sasson, Goldreich, Harsha, Sudan and Vadhan [SICOMP, 2006; CCC, 2005]. We then show how to preserve rectangularity under PCP composition and a smoothness-inducing transformation. This warrants refined and stronger notions of rectangularity, which we prove for the outer PCP and its transforms.

***Keywords*-Matrix Rigidity; PCP;**

I. Introduction

An $N \times N$ matrix with entries from a field $\mathbb{F}$ is said to be (Δ, ρ)-*rigid* if its Hamming distance from the set of $N \times N$ matrices of rank at most ρ is greater than Δ. In other words, an (Δ, ρ)-rigid matrix is a matrix that cannot be expressed as a sum of two matrices, $L + S$, where $\mathrm{rank}(L) \leq \rho$ and S has at most Δ non-zero entries. For concreteness, this work focuses on rigidity with respect to the field $\mathbb{F}_2$.

Constructing rigid matrices has been a long-standing open problem in computational complexity theory since their introduction by Valiant more than four decades ago [Val77]. Valiant showed that for any $(N^{1+\varepsilon}, N/\log\log(N))$-rigid matrix, evaluating the corresponding linear transformation requires circuits of either super-linear size or super-logarithmic depth. Thus, an explicit construction of a such matrices gives explicit problems that cannot be solved in linear-size logarithmic-depth circuits.

Razborov [Raz89] (see also [Wun12]) considered the other end of the spectrum of parameters, in which the distance Δ is quite high but the rank ρ is much smaller; namely, $\Delta = \delta \cdot N^2$ for constant $\delta > 0$ and $\rho = 2^{(\log\log N)^{\omega(1)}}$. Razborov showed that strongly-explicit matrices[1] with these rigidity parameters imply a lower bound for the communication-complexity analog of the Polynomial Hierarchy, PH^{cc}.

In other words, while Valiant's regime focuses on very high rank ρ, Razborov's focuses on very high distance Δ. Achieving lower bounds for either PH^{cc} (via Razborov's reduction) or linear-size log-depth circuits (via Valiant's reduction) are two central long-standing open questions in complexity theory.

Despite a lot of effort, state of the art results on matrix rigidity fall short of solving both Valiant and Razborov's challenges. The current best poly-time constructions yields $(\frac{N^2}{\rho}\log(\frac{N}{\rho}), \rho)$-rigid matrices, for any parameter ρ (see Friedman [Fri93]; Shokrollahi, Spielman, and Stemann [SSS97]). Goldreich and Tal [GT18] gave a randomized poly-time algorithm that uses $O(N)$ random bits and produces an $N \times N$ matrix that is $(\frac{N^3}{\rho^2 \log N}, \rho)$-rigid

†Work done while the first, second and third author were participating in the *Proofs, Consensus, and Decentralizing Society* program at the Simons Institute for the Theory of Computing, Berkeley, CA, USA.

‡Research supported by the Department of Atomic Energy, Government of India, under project no. 12-R&D-TFR-5.01-0500 and in part by the Swarnajayanti fellowship.

[1]An $N \times N$ matrix A is strongly-explicit if, given $i, j \in N$, one can compute $A_{i,j}$ in $\mathsf{polylog}(N)$ time.

2575-8454/20/$31.00 ©2020 IEEE
DOI 10.1109/FOCS46700.2020.00084

for any parameter $\rho \geq \sqrt{N}$, with high probability.[2] Recent breakthrough results [AW17], [DE19], [DL19] showed that several long-standing candidate construction of explicit matrices, like the Hadamard or the FFT matrices, are less rigid than previously believed, and in particular do not meet Valiant's challenge.

Most previous attempts were of combinatorial or algebraic nature (see a survey of Lokam [Lok09]). In contrast to these, a recent remarkable work of Alman and Chen [AC19] proposes a novel approach that uses ideas from complexity theory to construct rigid matrices in FNP:[3]

Theorem I.1 ([AC19])**.** *There exists a constant $\delta \in (0,1)$ such that for all $\varepsilon \in (0,1)$, there is an FNP-machine that, for infinitely many N, on input 1^N outputs an $N \times N$ matrix that is $(\delta \cdot N^2, 2^{(\log N)^{1/4-\varepsilon}})$-rigid.*[4]

Their result still does not attain the rank bounds required for Valiant's lower bounds, yet it vastly improves the state of the art of explicit rigid matrix constructions. On Razborov's end, the construction indeed meets the required rigidity parameters (in fact, greatly exceeds them), but does not fulfill the requirement of super-explicitness. That said, they use a tensoring argument to obtain $N \times N$ matrices still within Razborov's rigidity parameters, in which each entry is computable in non-deterministic time $2^{(\log \log N)^{\omega(1)}}$. While this is not super-explicit, it is an exponential improvement over previous results.

The surprising construction of Alman and Chen is a tour-de-force that ties together seemingly unrelated areas of complexity theory. A key area is the theory of Probabilistically Checkable Proofs (PCPs). PCPs provide a format of rewriting classical NP-proofs that can be efficiently verified based only on a small amount of random queries into the rewritten proof. The PCP Theorem [AS98], [ALMSS98] asserts that any NP-proof can be rewritten into a polynomially-longer PCP that can be verified using a constant number of queries. Alman and Chen make use of *efficient* and *short* PCPs for $\mathsf{NTIME}(2^n)$, as well as *smooth* PCPs. Momentarily, we too will make use of these properties, so let us give an informal description of these:

- *Efficient PCP:* A PCP for $\mathsf{NTIME}(T(n))$ is said to be *efficient* if the running time of the PCP verifier is sub-linear (or even logarithmic) in the length of the original NP-proof (i.e., $T(n)$).
- *Short PCP:* A PCP for $\mathsf{NTIME}(T(n))$ is said to be *short* if the length of the PCP is nearly linear (i.e., $T(N)^{1+o(1)}$) in the length of the original NP-proof (i.e., $T(n)$).
- *Smooth PCP:* A PCP is said to be *smooth* if for each input, every proof location is equally likely to be queried by the PCP verifier.

The PCP Theorem has had a remarkable impact on our understanding of the hardness of approximation of several combinatorial problems (see, e.g., a survey [Tre13]). Parallel to this line of work, Babai, Fortnow, Levin and Szegedy [BFLS91] initiated a long sequence of works [BSVW03], [BGHSV06], [BS08], [BGHSV05], [Din07], [MR08], [Mie09], [BV14], [Par20] that prove that there exist efficient, short and smooth PCPs for $\mathsf{NTIME}(2^n)$. Alman and Chen's construction makes use of these PCPs, as well as the non-deterministic time-hierarchy theorem [Zák83] and a fast (i.e., faster than $N^2/\mathsf{polylog}(N)$ time for $N \times N$ matrices) algorithm for counting the number of ones in a low-rank matrix [CW16].

A. Our results

Our work arises from asking if there exist PCPs with additional "nice" properties that can strengthen the above construction due to Alman and Chen. We answer this question in the affirmative, by (1) introducing a new variant of PCPs that we refer to as *rectangular PCPs*, (2) constructing efficient, short and smooth rectangular PCPs and (3) using these rectangular PCPs to strengthen and simplify the rigid-matrix construction in Theorem I.1. We begin by stating the improved rigid matrix construction.

Theorem I.2. *There is a constant $\delta \in (0,1)$ such that there is an FNP-machine that for infinitely many N, on input 1^N outputs an $N \times N$ matrix that is $(\delta \cdot N^2, 2^{\log N/\Omega(\log \log N)})$-rigid.*

We remark that Alman and Chen obtained a conditional result which proved a similar conclusion using the easy witness lemma of Impagliazzo, Kabanets and Wigderson [IKW02]: either $\mathsf{NQP} \not\subset \mathsf{P/poly}$ or for all $\varepsilon \in (0,1)$ there exists an FNP-algorithm that for infinitely many N, on input 1^N outputs an $N \times N$ matrix that is $(\delta \cdot N^2, 2^{(\log N)^{1-\varepsilon}})$-rigid. Our main result (Theorem I.2) is thus a common strengthening of both Theorem I.1 as well as this conditional result.

B. Rectangular PCP

Our result is obtained by a new notion of PCPs, called *rectangular PCPs*.[5] Briefly put, rectangular PCPs are PCPs

[2]Despite its "semi-explicitness", if this construction obtains Valiant's rigidity parameters, then lower bounds would be implied, since one can take the randomness to be part of the input, yielding a (related) explicit problem that has no linear-size logarithmic-depth circuits.

[3]The complexity class FNP is the function-problem extension of the decision-problem class NP. Formally, a relation $R(x,y)$ is in FNP if there exists a non-deterministic polynomial-time Turing machine M such that for any input x, $M(x)$ outputs y such $R(x,y) = 1$ or rejects if no such y exists.

[4]Their result is stated for $\mathsf{FP}^{\mathsf{NP}}$ but can be strengthened to FNP. More precisely, on infinitely many inputs 1^N that are accepted by the FNP-machine, any accepting path outputs a rigid matrix. Matrices obtained on different accepting paths may differ, but all of them are rigid. If one insists on outputting the same matrix on all accepting paths, then this can be done in $\mathsf{FP}^{\mathsf{NP}}$.

[5]We thank Ramprasad (RP) Saptharishi for suggesting the term "rectangular PCPs".

where the proofs are thought of as square matrices, and the random coins used by the verifier can be partitioned into two disjoint sets, one determining the *row* of each query and the other determining the *column*. To get a better feel for this new property, we examine the constraint satisfaction problem (CSP) underlying a rectangular PCP,[6] and defer the full definition to Section II.

Consider the classical NP-hard constraint satisfaction problem MAXCUT whose instance is a directed graph $G = (V, E)$ with n vertices and m edges and the goal is to find a subset $S \subseteq V$ that maximizes the number of edges cut (in either direction) between S and $V \setminus S$. The instance G is *rectangular* if the following condition is met.

- There exist two directed graphs G_1 and G_2 with $\ell = \sqrt{n}$ vertices and $r = \sqrt{m}$ edges each such that G is the product graph $G_1 \times G_2$, i.e., the edges of G satisfy the following *rectangular* property:
  - $((u_1, u_2), (v_1, v_2)) \in E(G)$ if and only if $(u_1, v_1) \in E(G_1)$ and $(u_2, v_2) \in E(G_2)$.

An instance G is said to be *τ-almost rectangular* for $\tau \in [0, 1)$ if G is the edge-disjoint union of m^τ product graphs $G_1^{(j)} \times G_2^{(j)}, j \in [m^\tau]$ where each of the product graphs $G_1^{(j)} \times G_2^{(j)}$ is defined on the same vertex set V and satisfies $|E(G_1^{(j)})| = |E(G_2^{(j)}| = m^{(1-\tau)/2}$. To distinguish rectangular graphs from almost-rectangular graphs, we will sometimes refer to them as perfectly rectangular.

This definition of rectangularity can be extended to arbitrary q-CSPs as follows. Let Φ be a q-CSP instance on a set V of $n = \ell^2$ variables. Let $\mathcal{C}$ be the set of $m = r^2$ constraints of Φ. As both the number of variables ($n = \ell^2$) and the number of constraints ($m = r^2$) are perfect squares, we will w.l.o.g. index them with double indices, $(i_i, i_2) \in [\ell] \times [\ell]$ and $(j_1, j_2) \in [r] \times [r]$. Let the (j_1, j_2)-th constraint in $\mathcal{C}$ involve the q variables $x_{c_1(j_1,j_2)}, \ldots, x_{c_q(j_1,j_2)}$. The instance Φ is said to be rectangular if for any $k \in [q]$, the address function $c_k : [r] \times [r] \to [\ell] \times [\ell]$ that specifies the k-th variable in the (j_1, j_2)-th clause can be decomposed into a product function $a_k \times b_k$ where $a_k, b_k \colon [r] \to [\ell]$. Almost rectangularity is defined similarly.

Back in the "proof systems" view, a PCP is said to be *(τ-almost) rectangular* if its underlying CSP is (τ-almost) rectangular. Thus, rectangularity can be viewed as natural structural property referring to the clause-variable relationship in the CSPs produced by the PCP. Our main technical result is that there exists an efficient, short, smooth and almost-rectangular PCP. A simplified version of our result is as follows; see full version for the exact statement [BHPT20].

Theorem I.3. *Let L be a language in $\mathsf{NTIME}(2^n)$. For every constants $s \in (0, 1/2)$ and $\tau \in (0, 1)$, there exists a constant-query, smooth and τ-almost rectangular PCP for L over the Boolean alphabet with perfect completeness, soundness error s, proof length at most $2^n \cdot \mathsf{poly}(n)$ and verifier running time at most $2^{O(\tau n)}$.*

[6] See, for example, [AB09, Chapter 18] for a description of the CSP underlying a PCP.

C. Rectangular PCPs to rigid matrices

We now sketch how rectangular PCPs can be used to construct rigid matrices. This will also serve as a motivation for the definition of rectangular PCPs. Our construction follows that of Alman and Chen (which fits within the *lower bounds from algorithms* framework of Williams [Wil13] – see Section I-E), with the main difference being the use of rectangular PCPs. We show that this simplifies their construction and improves the rigidity parameters it attains. The construction is inspired by the subtitular maxim:

Hard claims have complex proofs.

Informally speaking, the construction is an instantiation of this maxim where "complexity" refers to rigidity, "proofs" are PCPs, and "hard claims" are instances of the hard language guaranteed by the non-deterministic time hierarchy theorem (see next).

The main ingredients in our construction are as follows:

1) The non-deterministic time-hierarchy theorem [Zák83]: There exists a unary language $L \in \mathsf{NTIME}(2^n) \setminus \mathsf{NTIME}(2^n/n)$.
2) A non-trivial (i.e., sub-quadratic time) algorithm to compute the number of ones in a low-rank $\{0, 1\}$-valued matrix when given as input its low-rank decomposition $N = P \cdot Q$, where P and Q are matrices of dimensions $N \times \rho$ and $\rho \times N$, respectively. Such results with running time $N^{2-\varepsilon(\rho)}$ were developed by Chan and Williams [CW16] with $\varepsilon(\rho) = \Omega(1/\log \rho)$.
3) The existence of efficient, short, smooth and rectangular PCPs for $\mathsf{NTIME}(2^n)$, as guaranteed by Theorem I.3.

Let $L \in \mathsf{NTIME}(2^n) \setminus \mathsf{NTIME}(2^n/n)$ as guaranteed by the non-deterministic time-hierarchy theorem. By Theorem I.3, there exist efficient, short and smooth rectangular PCPs for L. Our goal is to show that either (a natural transformation of) the PCP yields a rigid matrix, or $L \in \mathsf{NTIME}(2^n/n)$ – a contradiction.

For simplicity of presentation, we will assume that the rectangular PCPs obtained in Theorem I.3 are perfectly rectangular and furthermore that the underlying CSP of the PCP is MAXCUT with completeness c and soundness s for some constants $0 < s < c < 1$. In other words, the PCP reduction reduces instances $x \in L$ to digraphs G which have a fractional cut of size at least c, and instances $x \notin L$ to digraphs G which do not have any cut of fractional size larger than s.

Let us understand what it means for the PCP to be short, smooth, efficient and rectangular: *"Rectangular"* refers to the fact that the digraph G is a product graph $G_1 \times G_2$;

"Short" implies that the size of G (i.e., the number of vertices and edges) is at most $r^2 = 2^n \cdot \mathsf{poly}(n)$; *"Smooth"* implies that the digraph G is regular (the degree of a vertex is the sum of its in-degree and out-degree); and *"Efficient and rectangular"* implies that for each of the two graphs G_1 and G_2, given an edge the vertices incident on the edge can be obtained in time $2^{\gamma n}$ (for a small constant $\gamma > 0$ of our choice).

For any instance x of the language L, any cut of the corresponding graph G is of the form $(S, V(G) \setminus S)$. Since $V(G) = V(G_1) \times V(G_2)$, we can identify the cut S with a $V(G_1) \times V(G_2)$-matrix with $\{0,1\}$ entries. Let $L_1, R_1 \in \{0,1\}^{E(G_1) \times V(G_1)}$ be the incidence matrices indicating the left and right endpoints of the edges in G_1 (i.e., if $e = (u,v) \in E(G_1)$ then $L_1(e,u) = R_1(e,v) = 1$). Similarly, define matrices $L_2, R_2 \in \{0,1\}^{E(G_2) \times V(G_2)}$. These matrices will not be computed explicitly; efficiency of the PCP implies that for any given row-index, the non-zero column of that row can be computed in time $2^{\gamma n}$. We will refer to this fact as the *somewhat-efficient* computation of these matrices.

Observe that the matrix $L_1 \cdot S \cdot L_2^T$ is an indicator matrix indicating if the left endpoint of the edge is in the set S or not. Similarly $R_1 \cdot S \cdot R_2^T$ refers to the indicator of the right endpoint of the edge. Hence, the matrix $M(S) := L_1 \cdot S \cdot L_2^T + R_1 \cdot S \cdot R_2^T$ is the indicator matrix of whether the edge is cut by the set S or not (with addition over $\mathbb{F}_2$). Thus, the size of the cut induced by the set S is exactly the number of ones in the matrix $M(S)$. Let us denote this quantity by $\mathrm{val}(S) := \#_1(M(S))$.

We will prove that for infinitely many $x \in L$, every cut $S^* \in \{0,1\}^{V(G_1) \times V(G_2)}$ of fractional size at least c is a $(\delta \cdot N^2, \rho)$-rigid matrix, for $\delta = (c-s)/3$ and $\rho = 2^{n/\Omega(\log n)}$. Assume towards contradiction that this was not the case. Then, there for long enough $x \in L$, there exists a cut S^* that is non-rigid. Since S^* is non-rigid, it is δ-close to some Boolean matrix $S = P \cdot Q$ such that P and Q are Boolean matrices of dimensions $V(G_1) \times \rho$ and $\rho \times V(G_2)$, respectively. We now make two observations.

- Since the cut S^* is of size at least c and is δ-close to S, it follows from the regularity of G that the cut induced by the set S is of size at least $c - 2\delta$. In other words, $\mathrm{val}(S) \geq c - 2\delta$.
- We can compute $\mathrm{val}(S) = \#_1(M(S))$ in time $O\left(r \cdot (\rho + 2^{\gamma n}) + r^{2-\varepsilon(2\rho)}\right)$ as follows. Recall that $M(S) = L_1 \cdot S \cdot L_2^T + R_1 \cdot S \cdot R_2^T$ and $S = P \cdot Q$. Hence,

$$\begin{aligned} M(S) &= L_1 \cdot P \cdot Q \cdot L_2^T + R_1 \cdot P \cdot Q \cdot R_2^T \\ &= \underbrace{\begin{pmatrix} L_1 & R_1 \end{pmatrix} \cdot \begin{pmatrix} P & 0 \\ 0 & P \end{pmatrix}}_{=:\widetilde{P}} \cdot \underbrace{\begin{pmatrix} Q & 0 \\ 0 & Q \end{pmatrix} \cdot \begin{pmatrix} L_2^T \\ R_2^T \end{pmatrix}}_{=:\widetilde{Q}} \end{aligned}$$

  So $M(S)$ is a matrix of rank at most 2ρ with a low-rank decomposition given by $M(S) = \widetilde{P} \cdot \widetilde{Q}$. Given matrices P and Q and the somewhat-efficient computation of the matrices L_1, L_2, R_1, R_2, the matrices $\widetilde{P}$ and $\widetilde{Q}$ may be computed in time $r \cdot (\rho + 2^{\gamma n})$. Finally, we invoke the algorithm of Chan and Williams to compute $\mathrm{val}(S) = \#_1(\widetilde{P} \cdot \widetilde{Q})$ in time $O\big(r^{2-\varepsilon(2\rho)}\big)$.

This suggests the following non-deterministic algorithm for checking membership in L

- On input 1^n
  1) Non-deterministically guess matrices $P \in \{0,1\}^{\ell \times \rho}$ and $Q \in \{0,1\}^{\rho \times \ell}$.
  2) Use the efficient PCP verifier to somewhat-efficiently compute the matrices L_1, L_2, R_1, R_2.
  3) Compute the matrices $\widetilde{P}$ and $\widetilde{Q}$.
  4) Compute the number of ones ν of the matrix $\widetilde{P} \cdot \widetilde{Q}$.
  5) Accept if and only if $\nu > s \cdot r^2$.

Indeed, this algorithm decides L: If $1^n \in L$, then there exists a guess $S = P \cdot Q$ that would get $\mathrm{val}(S) \geq (c - 2\delta) \cdot r^2 > s \cdot r^2$. On the other hand, if $1^n \notin L$, then by the soundness of the PCP, any cut S would have $\mathrm{val}(S) \leq s \cdot r^2$.

However, for a suitable choice of ρ, this algorithm runs in time $O\left(\ell \cdot \rho + r \cdot (\rho + 2^{\gamma n}) + r^{2-\varepsilon(2\rho)}\right) = O(2^n/n)$ – contradicting the time-hierarchy theorem. Hence, it is false that for every long enough $x \in L$, there exists a cut $S^* \in \{0,1\}^{V(G_1) \times V(G_2)}$ of fractional size at least c which is a non-rigid matrix. This immediately yields an FNP-algorithm that infinitely often outputs rigid matrices.

In the full version, we complete this sketch into a full proof that deals with two significant caveats: the PCP is only almost-rectangular, and the predicate of the PCP is not necessarily MAXCUT. The first is not a significant obstacle and the generalization is rather immediate. The second requires more care, but examining the proof reveals that we only used the fact that each clause has the same predicate or, in PCP jargon, that the predicate is *oblivious to the randomness*. To this end, we define a property of PCPs wherein the predicates have randomness-oblivious-predicates (ROP) and show that the rectangular PCPs constructed in Theorem I.3 can also be made ROP (see the full version for exact details).

Comparison with the Alman–Chen construction: Alman and Chen obtained a similar result conditioned on the assumption $\mathsf{NQP} \subseteq \mathsf{P/poly}$, using the easy witness lemma. To obtain an unconditional result, they used a bootstrapping argument which results in rigidity for rank at most $2^{(\log N)^{1/4-\varepsilon}}$. The above proof, on the other hand, is not conditioned on any assumption, does not require the easy witness lemma, and implies rigid matrices for rank $2^{\log N/\Omega(\log\log N)}$. In fact, there is almost no loss due to the PCPs in the above argument. For instance, if the number of ones in $N \times N$ matrices of rank $N^{0.999}$ could be computed in sub-quadratic time, then our construction would yield matrices rigid for rank $N^{0.99}$.

D. Constructing rectangular PCPs

The rectangular property of PCPs states that the underlying constraint satisfaction problem (CSP) has a product structure.

Warm-up: Rectangularity of some known constructions: As a warm-up, let us examine the rectangularity of some common PCP building blocks. The purpose of this warm-up is to become comfortable with the notion of rectangularity. Towards this end, we chose some simple examples from the PCP literature, rather than examples that are actually used in our construction.

First, it is immediate that PCPs obtained from *parallel repetition* are rectangular. Unfortunately, the size of these PCPs are far from being nearly-linear.

Next, recall the Blum–Luby–Rubinfeld (BLR) *linearity tester* [BLR93] that checks if a given function $f : \mathbb{F}_2^m \to \mathbb{F}_2$ is linear.

Algorithm (BLR Tester)**.** On oracle access to $f \colon \mathbb{F}_2^m \to \mathbb{F}_2$,

1) Sample $x, y \in_R \mathbb{F}_2^m$.
2) Query f at locations x, y, and $x + y$.
3) Accept if and only if $f(x) + f(y) + f(x + y) = 0$.

The (x, y)-th constraint in the above test queries the three locations $x, y, x + y \in \mathbb{F}_2^{3m}$. For even m, we can write $x = (x_1, x_2)$ and $y = (y_1, y_2)$ where $x_1, x_2, y_1, y_2 \in \mathbb{F}_2^{m/2}$. Thus, the $((x_1, x_2), (y_1, y_2))$-test queries the three locations (x_1, x_2), (y_1, y_2) and (x_1+y_1, x_2+y_2). Hence, the BLR test is perfectly rectangular. For similar reasons, the low-degree test (actually used in our construction) is also perfectly rectangular.

The actual construct: The warm-up gives us hope that PCPs constructed using from low-degree test are rectangular or can be made so with some modification. Our construction, essentially, realizes this hope. In particular we take a closer look at the short and efficient PCP construction of Ben-Sasson *et al*. [BGHSV06], [BGHSV05] and modify it suitably to obtain a rectangular PCP. This is a rather delicate operation and involves several subtleties along the way. We highlight the salient steps in the construction below.

For starters, recall another key ingredient in the construction of PCPs: the composition paradigm of Arora and Safra [AS98]. We will use the modular composition paradigm of Ben-Sasson *et al*. [BGHSV06] and Dinur and Reingold [DR06], wherein a *robust* PCP is composed with a PCP *of proximity*. Our construction of rectangular PCPs will proceed along the following lines.

1) We first show that the Reed–Muller based PCP construction due to Ben-Sasson *et al*. [BGHSV06], [BGHSV05] can be modified to yield a short almost-rectangular robust PCP. This involves a careful, step-by-step examination of this PCP. As indicated above, the low-degree component of this PCP is perfectly rectangular. However, this PCP also involves a sum-check component, which is inherently *not* rectangular, but is fortunately *almost* rectangular.
2) We then show that composition of an almost-rectangular robust PCP with a (not necessarily rectangular) PCP of proximity yields an almost-rectangular PCP.

Composing the outer robust PCP obtained in Item 1 with the short and efficient PCP of proximity of Mie [Mie09] yields a short, efficient and rectangular PCP with constant query complexity. However, this PCP is not necessarily smooth. By now, there are several standard techniques to "smoothify" a PCP in literature, but these techniques do not necessarily retain the rectangular property. To obtain a rectangular and smooth PCP, we actually work with a stronger notion of rectangularity, that we refer to as "rectangular-neighborhood-listing (RNL)" and show that a short and efficient PCP with RNL can be "smoothified" to yield the desired short, efficient, smooth and rectangular PCP.

E. Related Work

PCPs with structured queries: We view Theorem I.3 as continuing a line of work that explores the connection between the randomness of a PCP and the structure of its queries. A prominent advance in this direction is the work of Ben-Sasson and Viola [BV14]. They constructed short and efficient PCPs in which queries are a function of the input and a simple projection of the randomness (namely, a 1-*local* function: for a fixed input, each bit in each query location is a fixed bit of the randomness or its negation).[7] Although the query structure in [BV14] (and follow-up [Vio20]) is very simple, it is unclear whether their PCP is almost-rectangular or smooth—both playing a crucial role in our construction and its application.

In a different direction, Feige and Jozeph [FJ12] constructed PCPs in which the queries depend only on the randomness but not on the input. Recently, Austrin, Brown-Cohen, and Håstad [ABH19] improved this result to have optimal soundness error for certain verification predicates such as 3SAT and 3LIN.

Circuit Lower Bounds from Algorithms: The maxim "hard claims having complex proofs" is inspired by a result of Williams [Wil16], showing that witnesses for $\mathsf{NTIME}(2^n)\backslash\mathsf{NTIME}(2^n/n)$ cannot be truth-tables of certain small-size low-depth circuits (specifically, ACC^0 circuits). That work is a part of Williams's algorithmic approach to circuit lower bounds originating in [Wil13], [Wil14]. Roughly speaking, Williams's framework shows how to obtain lower bounds against a certain circuit class by designing non-trivial (i.e., better than exhaustive search)

[7] Interestingly, the construction of [BV14] is also an adaptation of a PCP from [BGHSV05], but not the same one as in our work. Namely, we build upon the Reed–Muller based PCP in [BGHSV05], whereas [BV14] builds upon the Reed–Solomon based PCP in the same paper.

SAT algorithms for circuits in the class. Williams [Wil13] also observed the usefulness of PCPs within this framework: using PCPs, one can obtain circuit lower bounds from any non-trivial derandomization.[8] Santhanam and Williams [SW14], Ben-Sasson and Viola [BV14], and Chen and Williams [CW19] further explored and tightened this connection. In this light, the overall proof strategy of Alman and Chen [AC19] can be seen as a surprising instantiation of Williams's framework for average-case hardness of the computational model of low-rank matrices.

Applications to Probabilistic Degree: Recently (and independently of this work), Viola [Vio20] showed the existence of functions on n variables in E^{NP} with approximate probabilistic degree $\Omega(n/\log^2 n)$ over $\mathbb{F}_2$, for infinitely many $n \in \mathbb{N}$. Using the known relation between matrix rigidity and approximate rank (see [AC19, Proposition 7.5]), Theorem I.2 implies a similar lower bound on the approximate probabilistic degree.

F. Organization

Our paper is organized as follows. All but the first two sections are omitted and appear in the full version.

- *Preliminaries (Section II).* We begin by giving a definitional treatment of PCPs and its variants. In particular, we formally define the rectangular PCP, which is the central object of our focus. We also define PCPs with randomness-oblivious-predicates (ROP), and briefly mention the rectangular-neighborhood-listing (RNL) property (discussed in detail in the full version).
- *From Rectangular PCPs to Rigid Matrices (Section III).* We show how the existence of efficient, short and smooth rectangular PCPs with ROP for $\mathsf{NTIME}(2^n)$ yields rigid matrices (thus proving Theorem I.2, modulus the actual rectangular PCP construction).
- *A Construction of Rectangular PCPs.* In the full version, we construct efficient, short and smooth rectangular PCPs for $\mathsf{NTIME}(T(n))$. The main steps in the construction are as follows:
  - We show how any PCP with RNL and ROP can be converted to a smooth and rectangular PCP with ROP. Hence, from this point onwards, we seek PCPs with RNL, rather than rectangular PCPs.
  - We show that the robust PCP verifier of Ben-Sasson *et al*. [BGHSV06], [BGHSV05] has RNL.
  - We show how to add ROP to any robust PCP with RNL.
  - We then show that any PCP of proximity, when composed with a robust PCP that has RNL and ROP, yields a PCP with RNL and ROP. Note that the PCP of proximity need not be rectangular.
  - Finally, we combine the results proved above to obtain our main construct: an efficient, short and smooth rectangular PCP (thus proving Theorem I.3).

[8]More precisely, from any non-trivial deterministic estimation of the acceptance probability of a circuit, up to a constant additive error.

II. PCPs: definitions and variants

The main focus of this section is to introduce the notion of *rectangular PCPs*, the central object of interest in this work. To this end, we begin by recalling the definition of PCPs and smooth PCPs, before proceeding to define rectangular PCPs.

Notation: Let Σ be any finite alphabet. For $u, v \in \Sigma^n$, the relative Hamming distance between u and v, denoted by $\delta(u,v)$, is the fraction of locations on which u and v differ (i.e., $\delta(u,v) \coloneqq |\{i : u_i \neq v_i\}|/n$). We say that u is δ-*close* to v (resp., δ-*far* from v) if $\delta(u,v) \leq \delta$ (resp., $\delta(u,v) > \delta$). The relative distance of a string u to a set V of strings is defined as $\delta(u,V) \coloneqq \min_{v \in V}\{\delta(u,v)\}$.

A. Standard PCPs

We begin by recalling the formalism of an efficient PCP verifier. As is standard in this literature, we restrict our attention to *non-adaptive* verifiers.

Definition II.1 (efficient PCP verifiers)**.**

- *Let $r, q, m, d, t, \sigma \colon \mathbb{N} \to \mathbb{N}$. A (r,q,m,d,t)-*restricted verifier *over alphabet $\Sigma \coloneqq \{0,1\}^\sigma$ is a probabilistic algorithm V that, on an input x of length n, tosses $r \coloneqq r(n)$ random coins R and generates a sequence of $q \coloneqq q(n)$* query locations *$I \coloneqq (i^{(1)}, \ldots, i^{(q)})$, where each $i^{(k)} \in [m(n)]$, and a* (decision) predicate *$D \colon \Sigma^q \to \{0,1\}$ of size at most $d(n)$ in time at most $t(n)$.*
  Think of V as representing a probabilistic oracle machine that queries the proof oracle $\pi \in \Sigma^m$, for the positions in I, receives the q symbols $\pi|_I \coloneqq (\pi_{i^{(1)}}, \ldots, \pi_{i^{(q)}})$, and accepts iff $D(\pi|_I) = 1$.
- *We write $(I, D) \xleftarrow{R} V(x)$ to denote the queries and predicate generated by V on input x and random coin tosses. To explicitly mention the random coins R, we write $(I, D) \leftarrow V(x; R)$.*
- *We call r the* randomness complexity*, q the* query complexity*, m the* proof length*, d the* decision complexity *and t the* running time *of V. σ is called the* answer complexity *of V, and will usually be omitted.*[9]

It will be convenient at times to have the following graphical description of the verifier. Given a (r,q,m,d,t)-restricted verifier and input x, consider the bipartite graph $G(V,x) \coloneqq (L = \{0,1\}^r, R = [m], E)$ where $(R,i) \in E$ if the verifier V on input x and random coins R queries location i in the proof. Clearly, the graph $G(V,x)$ is q-left regular.

[9]All PCPs in this work will be Boolean (i.e., $\sigma = 1$), except for an intermediate PCP that appears in the full version. Even there, it will be more convenient to consider the alphabet size $|\Sigma| = 2^\sigma$ rather than σ.

We can now define the standard notion of efficient PCPs with perfect completeness.

Definition II.2 (PCP)**.** *For a function $s : \mathbb{N} \to [0,1]$, a verifier V is a* probabilistically checkable proof system (PCP) *for a language L with* soundness error *s if the following two conditions hold for every string x:*

- Completeness: *If $x \in L$ then there exists π such that $V(x)$ accepts oracle π with probability 1. Formally,*

$$\exists \pi \qquad \Pr_{(I,D) \xleftarrow{R} V(x)} [D(\pi|_I) = 1] = 1.$$

- Soundness: *If $x \notin L$ then for every oracle π, the verifier $V(x)$ accepts π with probability strictly less than s. Formally,*

$$\forall \pi \qquad \Pr_{(I,D) \xleftarrow{R} V(x)} [D(\pi|_I) = 1] < s(|x|).$$

By now, we know of several such efficient PCP constructions, one of which we state below.

Theorem II.3 (efficient PCPs for $\mathsf{NTIME}(T)$ [BGHSV05, Theorem 2.6])**.** *Suppose that L is a language in $\mathsf{NTIME}(T(n))$ for some non-decreasing function $T \colon \mathbb{N} \to \mathbb{N}$. Then for every $\varepsilon \in (0,1)$, L has a PCP verifier over $\{0,1\}$ with soundness error ε, query complexity $O(1/\varepsilon)$ and randomness complexity $\log T(n) + \log^{O(\varepsilon)} T(n)$.*

While constructing variants of the above PCP, we will particularly be interested in smooth PCPs.

Definition II.4 (smooth PCP)**.** *Given a (r, q, m, d, t)-restricted verifier V, an input x and $i \in [m]$, let $Q_x(i)$ denote the probability with which the verifier V outputs i on a random query $k \in [q]$. Formally,*

$$Q_x(i) \coloneqq \Pr_{R, k \in [q]} [i^{(k)} = i | (I, D) \leftarrow V(x; R)].$$

The PCP verifier V is said to be smooth if for all $i, j \in [m]$, $Q_x(i) = Q_x(j)$.

Thus, smooth PCPs refer to PCPs whose verifiers query all locations of the proof oracle equally likely (or equivalently in the above graphical description, verifiers whose corresponding bipartite graphs are also right-regular).[10]

Remark II.5 (tolerance of smooth PCPs)**.** *A smooth PCP is* tolerant *of errors in a correct proof, in the sense that a proof that is close to a correct one is accepted with good probability. Concretely, suppose V makes q queries to its proof and is smooth. Then if π is a correct proof for V (i.e. accepted w.p. 1) and π^* is δ-close to π in relative Hamming distance, then π^* is accepted with probability at least $1 - q \cdot \delta$.*

The PCPs constructed in Theorem II.3 are not necessarily smooth, however they can be made smooth without too much of an overhead. In this work we will be interested in *smoothening* the PCP maintaining yet another property, *rectangularity*, which we introduce in the following section.

B. Rectangular PCPs

We now define *rectangular PCPs*, the central object of interest in this work. As the name suggests, rectangular PCPs are PCPs in which the proof oracle $\pi : [m] \to \Sigma$, an m-length string, is interpreted as a matrix $\pi : [\ell] \times [\ell] \to \Sigma$ for some ℓ such that $m = \ell^2$ (yes, we assume that the proof lengths are always squares of integers). Furthermore, the verifier is also "rectangular" in the sense that the randomness $R \in \{0,1\}^r$ is also partitioned into 2 parts $R = (R_{\mathrm{row}}, R_{\mathrm{col}})$ such that the row index of the queries is obtained from the "row randomness" R_{row} while the column index of the queries is obtained from the "column randomness" R_{col}.

The above informal description assumes "perfect" rectangularity while the definition below allows for the relaxed notion of "almost-rectangularity", in which randomness is partitioned into three parts: row and column (as above), as well as a small *shared* part that is used for obtaining both the rows and the columns of the queries.

Definition II.6 (Rectangular PCP)**.** *For $\tau \in [0,1)$, a (r, q, ℓ^2, d, t)-restricted verifier V is said to be τ-*rectangular *if the following holds.*

The random coin tosses $R \in \{0,1\}^r$ can be partitioned into 3 parts

$$R = (R_{\mathrm{row}}, R_{\mathrm{col}}, R_{\mathrm{shared}}) \in \{0,1\}^{(1-\tau)r/2} \times \{0,1\}^{(1-\tau)r/2} \times \{0,1\}^{\tau r},$$

such that the verifier V on input x of length n and random coins R produces a sequence of q proof locations *$I = ((i^{(1)}_{\mathrm{row}}, i^{(1)}_{\mathrm{col}}), \ldots, (i^{(q)}_{\mathrm{row}}, i^{(q)}_{\mathrm{col}}))$ as follows:*

- $I_{\mathrm{row}} \coloneqq (i^{(1)}_{\mathrm{row}}, \ldots, i^{(q)}_{\mathrm{row}}) = V_{\mathrm{row}}(x; R_{\mathrm{row}}, R_{\mathrm{shared}})$,
- $I_{\mathrm{col}} \coloneqq (i^{(1)}_{\mathrm{col}}, \ldots, i^{(q)}_{\mathrm{col}}) = V_{\mathrm{col}}(x; R_{\mathrm{col}}, R_{\mathrm{shared}})$,
- *Generating I_{row}, I_{col}, and the decision predicate*[11] *take a total of at most $t(n)$ time.*

In other words, the row (respectively column) indices of the queries are only a function of the row (respectively column) and shared parts of the randomness. If $\tau = 0$, we will say the verifier V is perfectly rectangular, and otherwise V is almost rectangular. When it is obvious from context, we will say that V is simply rectangular, omitting the "τ-" qualifier.

[10] *Minor historical inconsistencies in the definition of smoothness:* Several previous works [KT00], [Par20] defined a smooth oracle machine as one in which each location of the oracle has equal probability of being queried by the machine *in any of its queries* (rather than in a random query, as in Definition II.4 as well as other prior works [GS06], [BGHSV06]). Indeed, both definitions are equivalent assuming the machine never queries the same location twice for any given random coin sequence R. Our definition is more convenient as it coincides with right-regularity of the corresponding bipartite graph even without this assumption.

[11] It is natural to wonder how the decision predicate depends on the randomness. This is considered in Section II-D.

C. Rectangular Neighbor-Listing (RNL)

A careful reading of the construction of PCPs mentioned in Theorem II.3 will reveal that they are in fact rectangular. However, for our application, we will need rectangular PCPs that are also smooth. Later, we will "smoothen" a PCP while maintaining its rectangularity (see the full version), for which we need a stronger property that we refer to as *rectangular-neighbor-listing (RNL)*. Briefly, a PCP has RNL if, for any location in the proof i, it is possible to list all configurations (of random coins and query index) that result in a query to the ith location. A formal definition of RNL appears in the full version, in which we also show that any PCP with RNL can be made *smooth* and *rectangular*.

D. Randomness-oblivious predicates (ROP)

For our application of rectangular PCPs (Section III), we would like the decision circuit to depend only on the shared part of the randomness. However, we do not know how to obtain such a PCP (that is also *smooth* and *short*), so we allow the decision circuit to take a limited number of *parity checks* of the entire randomness. Like the decision circuit, the choice of parity checks depends only on the shared part of the randomness.

Definition II.7 (efficient PCP verifiers with τ-ROP)**.** *For $\tau \in [0,1)$, a (r,q,m,d,t)-restricted verifier V is said to have the τ-*randomness-oblivious predicates *(τ-ROP) if the following holds.*

The random coin tosses $R \in \{0,1\}^r$ can be partitioned into two parts

$$R = (R_{\text{obliv}}, R_{\text{aware}}) \in \{0,1\}^{(1-\tau)r} \times \{0,1\}^{\tau r},$$

such that the verifier V on input x of length n and random coins R runs in time $t(n)$, and

1) *Based only on R_{aware}:*
   a) *Constructs a* (decision) predicate *$D \leftarrow V(x; R_{\text{aware}})$ of size at most $d(n)$.*
   b) *Constructs a sequence of* randomness parity checks *$(C_1, \ldots, C_p) \leftarrow V(x; R_{\text{aware}})$, each of which is an affine function on $\{0,1\}^{(1-\tau)r} \to \{0,1\}$.*[12]
2) *Based on all of the randomness $R = (R_{\text{obliv}}, R_{\text{aware}})$, produces a sequence of q* proof locations *$I = (i^{(1)}, \ldots, i^{(q)})$, where each $i^{(k)} \in [m(n)]$.*

Think of V as representing a probabilistic oracle machine that queries proof oracle π and gets answer symbols $\pi|_I$, computes parity checks $P \coloneqq (C_1(R_{\text{obliv}}), \ldots, C_p(R_{\text{obliv}}))$ and accepts iff $D(\pi|_I, P) = 1$.
We write $(I, P, D) \xleftarrow{R} V(x)$ to denote the queries, predicate, and parities generated by V on input x. To explicitly mention the random coins R, we write $(I, P, D) \leftarrow V(x; R)$.

[12]These are affine functions of the oblivious part only, but they encompass parities on *all* of the randomness by including the parity of the aware part in the constant term.

We call p the parity-check complexity *of V.*

We view ROP as a secondary property to RNL and rectangularity, and for simplicity we sometimes omit it from informal discussions (e.g., the title Section III). Indeed, in the full version we show a simple way of adding adding ROP to any PCP while essentially increasing only its decision complexity.

III. From rectangular PCPs to rigid matrices

Alman and Chen [AC19] show how to construct rigid matrices using efficient, short and smooth PCPs. In this section, we show how efficient, short, smooth and *rectangular* PCPs can be used to obtain a simpler and stronger construction of rigid matrices.

Lemma III.1. *Let $\tau \in (0,1)$ and $\rho : \mathbb{N} \to \mathbb{N}$. Let $L \in \mathsf{NTIME}(2^n) \setminus \mathsf{NTIME}(O(2^n/n))$ be a unary language. Suppose L has a PCP with soundness error s and a $\big(r, q, \ell^2, d, t\big)$-restricted verifier V over alphabet $\{0,1\}$ with $\ell(n) = \mathsf{poly}(2^n)$ strictly monotone increasing. Assume further that V is τ-rectangular and has τ-ROP with parity-check complexity p, and that the* shared *and the* aware *parts of the randomness are the same. Lastly, assume that the following inequalities hold:*

1) $\frac{1+\tau}{2} \cdot r + \log(t+\rho) \le n - \log n$.
2) $q + p + r - \Omega\Big(\frac{(1-\tau)r}{\log((q+p)\rho)}\Big) \le n - \log n$.

Then, there is an FNP*-machine such that, for infinitely many $N \in \mathbb{N}$, on input 1^N, outputs an $N \times N$ matrix with $\mathbb{F}_2$ entries which is $\Big(\frac{1-s}{q} \cdot N^2, \rho(\ell^{-1}(N))\Big)$-rigid.*

To prove Lemma III.1, we make use of the following fast algorithm that counts the number of 1's in a low rank matrix (given its low rank decomposition).

Theorem III.2 ([CW16], [AC19])**.** *Given two matrices $A \in \mathbb{F}_2^{n \times \rho}$ and $B \in \mathbb{F}_2^{\rho \times n}$ where $\rho = n^{o(1)}$, there is a (deterministic) algorithm that computes the number of 1's in the matrix $A \cdot B$ in time $T(n,\rho) \coloneqq n^{2-\Omega\left(\frac{1}{\log \rho}\right)}$.*

We now prove Lemma III.1.

Proof of Lemma III.1: Let V be the postulated PCP verifier for L. The FNP-machine computing rigid matrices (infinitely often) runs as follows. On input 1^N,

1) If $N = \ell(n)$ for some $n \in \mathbb{N}$: [13]
   a) Guess an $N \times N$ matrix denoted by π.
   b) Emulate $V^\pi(1^n)$ on all possible 2^r random coins. If V accepted on all randomness, accept and output π; else, reject.
2) Else ($N \neq \ell(n)$ for any n), reject.

This machine runs in time $O(2^r \cdot t) = \mathsf{poly}(2^n) = \mathsf{poly}(\ell(n)) = \mathsf{poly}(N)$, and whenever $N = \ell(n)$ for some

[13]This can be done in time $O(n) = O(\log N)$ by finding the first integer n such that $\ell(n) \ge N$, and verifying that $\ell(n) = N$.

n such that $1^n \in L$, one of its non-deterministic guesses lead to acceptance by completeness of the PCP. Note that there could be multiple non-deterministic guesses that lead to acceptance. We show that for infinitely many $N = \ell(n)$ such that $1^n \in L$, any guessed π that leads to acceptance is $\left(\frac{1-s}{q} \cdot N^2, \rho\right)$-rigid for $\rho := \rho(n) = \rho\big(\ell^{-1}(N)\big)$.

Assume towards contradiction that this is not the case. Then, there exists an n_0 such that for any $n \geq n_0$, $1^n \in L$ if and only if there exists a proof π (for the verifier V) which is $\frac{1-s}{q}$-close to a rank ρ matrix. We describe a non-deterministic algorithm that decides L in time $O(2^n/n)$ – a contradiction. Given input 1^n,

1) Guess matrices A and B of dimensions $\ell \times \rho$ and $\rho \times \ell$ respectively. (The right guess is when $A \cdot B$ is δ-close to π; by smoothness, the acceptance probability of $A \cdot B$ will then be close to that of π, and the task is reduced to estimating the acceptance probability of $A \cdot B$.)
2) Compute the acceptance probability of $A \cdot B$ by V as follows. For each sequence of coins in the shared part of the randomness $R_{\text{shared}} \in \{0,1\}^{\tau \cdot r}$:
   a) Compute the predicate $D := D(R_{\text{shared}})$ and randomness parity checks $C_j := C_j(R_{\text{shared}})$, $j \in [p]$.
   b) *Prepare queries into proof:* For each $k \in [q]$,
      i) *Prepare left matrices:* Compute the $2^{(1-\tau)r/2} \times \rho$ matrix $A^{(k)}$ whose R_{row}-th row is just the row indexed by $i^{(k)}_{\text{row}}(R_{\text{row}}, R_{\text{shared}})$ in A, for any $R_{\text{row}} \in \{0,1\}^{(1-\tau)r/2}$.
      ii) *Prepare right matrices:* Compute the $\rho \times 2^{(1-\tau)r/2}$ matrix $B^{(k)}$ whose R_{col}-th column is just the column indexed by $i^{(k)}_{\text{col}}(R_{\text{col}}, R_{\text{shared}})$ in B, for any $R_{\text{col}} \in \{0,1\}^{(1-\tau)r/2}$.

      Observe that $\big(A^{(k)} \cdot B^{(k)}\big)_{R_{\text{row}}, R_{\text{col}}}$ is exactly the k-th bit read by the verifier on randomness $(R_{\text{row}}, R_{\text{col}}, R_{\text{shared}})$.
   c) *Prepare randomness parity checks:* For each $j \in [p]$,
      i) Compute the $2^{(1-\tau)r/2} \times 3$ matrix $A^{(q+j)}$ and the $3 \times 2^{(1-\tau)r/2}$ matrix $B^{(q+j)}$, for which $\big(A^{(q+j)} \cdot B^{(q+j)}\big)_{R_{\text{row}}, R_{\text{col}}} = C_j(R_{\text{row}}, R_{\text{col}})$. Such matrices exist and can be computed in time $O(r \cdot 2^{(1-\tau)r/2})$, as described in the full version.
   d) *Fast counting:* Fourier analysis tells us that there are (unique) coefficients $\left\{\widehat{D}(K)\right\}_{K \subseteq [q+p]}$ such that for any $y \in \{0,1\}^{q+p}$,

$$D(y_1, \ldots, y_{q+p}) = \sum_{K \subseteq [q+p]} \widehat{D}(K)(-1)^{\bigoplus_{i \in K} y_i}.$$

In particular, by linearity of expectation, to compute the expected value (which is also the acceptance probability) of $D(y)$ over a random y sampled from some distribution, it suffices to compute the expected value of all parity predicates on over y, namely $\mathbb{E}_y\big[\bigoplus_{i \in K} y_i\big]$ for all $K \subseteq [q+p]$.

This observation is useful because the final task is to compute the acceptance probability of the predicate D on inputs $\big(A^{(1)} \cdot B^{(1)}\big)_{R_{\text{row}}, R_{\text{col}}}, \ldots, \big(A^{(q+p)} \cdot B^{(q+p)}\big)_{R_{\text{row}}, R_{\text{col}}}$ for uniformly random R_{row} and R_{col}. Thus, it suffices to compute the acceptance probability of all parity predicates, i.e., the number of 1's in $\bigoplus_{k \in K} A^{(k)} \cdot B^{(k)}$ for each $K \subseteq [q+p]$.

For each K, note that $\bigoplus_{k \in K} A^{(k)} \cdot B^{(k)}$ is the product of a $2^{(1-\tau)r/2} \times (|K|\rho)$ and a $(|K|\rho) \times 2^{(1-\tau)r/2}$ matrix over $\mathbb{F}_2$ (namely, concatenate the rows of the $|K|$ matrices $\{A^{(k)}\}_{k \in K}$ and concatenate the columns of the $|K|$ matrices $\{B^{(k)}\}_{k \in K}$). Thus, for each K, computing the acceptance probability of $\bigoplus_{k \in K} A^{(k)} \cdot B^{(k)}$ can be done with the fast counting algorithm for low-rank matrices of Theorem III.2. Its runtime is

$$\begin{aligned} &T(2^{(1-\tau)r/2}, (q+p) \cdot \rho) \\ &\quad = \left(2^{(1-\tau)r/2}\right)^{\left(2-\Omega\left(\frac{1}{\log((q+p)\cdot\rho)}\right)\right)} \\ &\quad = 2^{(1-\tau)r - \Omega\left(\frac{(1-\tau)r}{\log((q+p)\cdot\rho)}\right)} \end{aligned}$$

3) We have thus computed the acceptance probability of $A \cdot B$ by the verifier V. If this probability is at least the soundness error s, decide that the input 1^n is in L. Otherwise, decide that the input is not in L.

We claim that the algorithm correctly decides L, for input length $n \geq n_0$. Indeed, if $1^n \in L$ then, when guessing A and B such that $A \cdot B$ is $(\frac{1-s}{q})$-close to the correct proof π for 1^n (such A and B exist by our assumption towards contradiction), smoothness of the verifier V implies that its acceptance probability is at least $1 - q \cdot (\frac{1-s}{q}) = s$. On the other hand, if $1^n \notin L$, then soundness of V implies any guessed A and B leads to rejection with probability strictly less than s. Since n_0 is constant we can hard-wire the values of L on 1^n for $n < n_0$ so that the algorithm correctly decides L on all inputs.

As for its runtime, observe that Item 2a takes time $O(t)$, Item 2b takes time $O(2^{(1-\tau)r/2} \cdot (t+\rho))$ and Item 2c takes time $O(r \cdot 2^{(1-\tau)r/2})$. Since $t \geq r$, these are dominated by $O(2^{(1-\tau)r/2} \cdot (t+\rho))$. Therefore the runtime of the algorithm is

$$O\Big(2^{\frac{(1+\tau)r}{2}} \cdot (t+\rho) \;+\; 2^{q+p+r-\Omega\left(\frac{(1-\tau)r}{\log((q+p)\cdot\rho)}\right)}\Big).$$

By the assumption on the parameters of the PCP, this is at most $O(2^n/n)$ – a contradiction. ∎

Now that we formalized a connection between rectangular PCPs and rigid matrices, let us introduce the PCP that we construct in the full version, and show how it implies the rigid matrix construction asserted in Theorem I.2.

Theorem III.3. *For any $L \in \mathsf{NTIME}(2^n)$, and constants $s \in (0, 1/2)$ and $\tau \in (0, 1)$, L has a PCP verifier over alphabet $\{0, 1\}$ with the following parameters:*

- *Randomness complexity $r(n) = n + O(\log n)$*
- *Proof length $\ell(n)^2 = 2^n \cdot \mathsf{poly}(n)$.*
- *Soundness error s.*
- *Decision, query and parity-check complexities all $O(1)$.*
- *Verifier runtime $t(n) = 2^{O(\tau n)}$.*
- *The verifier is τ-rectangular and has τ-ROP. Furthermore, the shared and the aware parts of the randomness is the same.*

Next, we restate the rigid matrix construction asserted in Theorem I.2 and reckon that it is obtained by combining Theorem III.3 and Lemma III.1.

Corollary III.4 (Theorem I.2, restated)**.** *There is a constant $\delta \in (0, 1)$ such that there is an FNP-machine that for infinitely many N, on input 1^N outputs an $N \times N$ matrix that is $(\delta \cdot N^2, 2^{\log N/\Omega(\log\log N)})$-rigid.*

Proof: From [Zák83], there exists a unary language $L \in \mathsf{NTIME}(2^n) \setminus \mathsf{NTIME}(O(2^n/n))$. Let L be any such language. Fix $\rho(n) := 2^{n/(K \log n)}$ for a large enough constant K to be determined later. We verify that the parameters of the PCP of Theorem III.3, for a sufficiently small $\tau \in (0, 1)$, satisfy all the conditions from Lemma III.1 for this ρ. Let q and p denote the query complexity and parity-check complexity of the PCP respectively. Set $\delta = (1 - s)/q$. Now, for small enough τ,

$$\begin{aligned}\left(\tfrac{1+\tau}{2}\right) \cdot r + \log(\rho + t) \\ \leq (1/2 + O(\tau))n + O(n/\log n) < n - \log n,\end{aligned}$$

as required in Item 1. Also, the parameters satisfy Item 2, because

$$\begin{aligned}q + p + r - \Omega\left(\frac{(1-\tau)r}{\log((q+p)\rho)}\right) \\ \leq n + O(\log n) - \Omega(K \log n) < n - \log n,\end{aligned}$$

where the last inequality holds for a suitable choice of the constant K. As the proof length is $\ell(n)^2 = 2^n \cdot \mathsf{poly}(\log n)$, we have $\ell^{-1}(N) = \Theta(\log N)$. Therefore, $\rho(\ell^{-1}(N)) = 2^{\log N/\Omega(\log\log N)}$ and the corollary follows from Lemma III.1. ∎

Remark III.5. *The only bottleneck preventing Lemma III.1 from giving rigid matrices for polynomial ranks, $\rho = N^{\Omega(1)}$, via Theorem III.3 is the runtime of the counting algorithm used in Item 2d. That is, such results would be obtained if there was an algorithm for counting the number of nonzero entries in a rank $N^{\Omega(1)}$ matrix (of dimensions $N \times N$) that ran in time slightly better than $O(N^2)$. Our reduction would go through even if the algorithm's answer was only* approximately *correct (while losing the respective approximation factor in the distance of the resulting matrices from low rank ones). In fact, this seems to be the only bottleneck even up to rank $\rho = N^{1-O(\tau)}$.*

Acknowledgment

We thank Lijie Chen and Emanuele Viola for their helpful comments on Section I-E.

References

[AB09] Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, 2009.

[ABH19] Per Austrin, Jonah Brown-Cohen, and Johan Håstad. *Optimal inapproximability with universal factor graphs*, 2019. (manuscript). `ecccc:2019/TR19-151`.

[AC19] Josh Alman and Lijie Chen. *Efficient construction of rigid matrices using an NP oracle*. In *Proc. 60th IEEE Symp. on Foundations of Comp. Science (FOCS)*, pages 1034–1055. 2019.

[ALMSS98] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. *Proof verification and the hardness of approximation problems*. J. ACM, 45(3):501–555, May 1998. (Preliminary version in *33rd FOCS*, 1992). `ecccc:1998/TR98-008`.

[AS98] Sanjeev Arora and Shmuel Safra. *Probabilistic checking of proofs: A new characterization of NP*. J. ACM, 45(1):70–122, January 1998. (Preliminary version in *33rd FOCS*, 1992).

[AW17] Josh Alman and R. Ryan Williams. *Probabilistic rank and matrix rigidity*. In *Proc. 49th ACM Symp. on Theory of Computing (STOC)*, pages 641–652. 2017. `arXiv:1611.05558`.

[BFLS91] László Babai, Lance Fortnow, Leonid A. Levin, and Mario Szegedy. *Checking computations in polylogarithmic time*. In *Proc. 23rd ACM Symp. on Theory of Computing (STOC)*, pages 21–31. 1991.

[BGHSV05] Eli Ben-Sasson, Oded Goldreich, Prahladh Harsha, Madhu Sudan, and Salil Vadhan. *Short PCPs verifiable in polylogarithmic time*. In *Proc. 20th IEEE Conf. on Comput. Complexity*, pages 120–134. 2005. Full version available at http://www.tcs.tifr.res.in/~prahladh/papers/BGHSV2/BGHSV2005.pdf.

[BGHSV06] ———. *Robust PCPs of proximity, shorter PCPs and applications to coding*. SIAM J. Comput., 36(4):889–974, 2006. (Preliminary version in *36th STOC*, 2004). `ecccc:2004/TR04-021`.

[BHPT20] AMEY BHANGALE, PRAHLADH HARSHA, ORR PARADISE, and AVISHAY TAL. *Rigid matrices from rectangular PCPs or Hard Claims have Complex Proofs*, 2020. (manuscript). arXiv:2005.03123, eccc:2020/TR20-075.

[BLR93] MANUEL BLUM, MICHAEL LUBY, and RONITT RUBINFELD. *Self-testing/correcting with applications to numerical problems*. J. Comput. Syst. Sci., 47(3):549–595, December 1993. (Preliminary version in *22nd STOC*, 1990).

[BS08] ELI BEN-SASSON and MADHU SUDAN. *Short PCPs with polylog query complexity*. SIAM J. Comput., 38(2):551–607, 2008. (Preliminary version in *37th STOC*, 2005). eccc:2004/TR04-060.

[BSVW03] ELI BEN-SASSON, MADHU SUDAN, SALIL VADHAN, and AVI WIGDERSON. *Randomness-efficient low degree tests and short PCPs via epsilon-biased sets*. In *Proc. 35th ACM Symp. on Theory of Computing (STOC)*, pages 612–621. 2003.

[BV14] ELI BEN-SASSON and EMANUELE VIOLA. *Short PCPs with projection queries*. In JAVIER ESPARZA, PIERRE FRAIGNIAUD, THORE HUSFELDT, and ELIAS KOUTSOUPIAS, eds., *Proc. 42nd International Colloq. of Automata, Languages and Programming (ICALP), Part I*, volume 8572 of *LNCS*, pages 163–173. Springer, 2014. eccc:2014/TR14-017.

[CW16] TIMOTHY M. CHAN and R. RYAN WILLIAMS. *Deterministic APSP, orthogonal vectors, and more: Quickly derandomizing razborov-smolensky*. In ROBERT KRAUTHGAMER, ed., *Proc. 27th Annual ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 1246–1255. 2016.

[CW19] LIJIE CHEN and R. RYAN WILLIAMS. *Stronger connections between circuit analysis and circuit lower bounds, via PCPs of proximity*. In *Proc. 34th Comput. Complexity Conf.*, volume 137 of *LIPIcs*, pages 19:1–19:43. Schloss Dagstuhl, 2019.

[DE19] ZEEV DVIR and BENJAMIN L. EDELMAN. *Matrix rigidity and the Croot-Lev-Pach lemma*. Theory Comput., 15(8):1–7, 2019. arXiv:1708.01646.

[Din07] IRIT DINUR. *The PCP theorem by gap amplification*. J. ACM, 54(3):12, 2007. (Preliminary version in *38th STOC*, 2006). eccc:2005/TR05-046.

[DL19] ZEEV DVIR and ALLEN LIU. *Fourier and circulant matrices are not rigid*. In *Proc. 34th Comput. Complexity Conf.*, volume 137 of *LIPIcs*, pages 17:1–17:23. Schloss Dagstuhl, 2019. arXiv:1902.07334, eccc:2019/TR19-129.

[DR06] IRIT DINUR and OMER REINGOLD. *Assignment testers: Towards a combinatorial proof of the PCP Theorem*. SIAM J. Comput., 36:975–1024, 2006. (Preliminary version in *45th FOCS*, 2004).

[FJ12] URIEL FEIGE and SHLOMO JOZEPH. *Universal factor graphs*. In ARTUR CZUMAJ, KURT MEHLHORN, ANDREW M. PITTS, and ROGER WATTENHOFER, eds., *Proc. 39th International Colloq. of Automata, Languages and Programming (ICALP), Part I*, volume 7391 of *LNCS*, pages 339–350. Springer, 2012. arXiv:1204.6484.

[Fri93] JOEL FRIEDMAN. *A note on matrix rigidity*. Combinatorica, 13(2):235–239, 1993.

[GS06] ODED GOLDREICH and MADHU SUDAN. *Locally testable codes and PCPs of almost linear length*. J. ACM, 53(4):558–655, 2006. (Preliminary version in *43rd FOCS*, 2002). eccc:2002/TR02-050.

[GT18] ODED GOLDREICH and AVISHAY TAL. *Matrix rigidity of random Toeplitz matrices*. Comput. Complexity, 27(2):305–350, 2018. Preliminary version in *48th STOC*, 2002). eccc:2015/TR15-079.

[IKW02] RUSSELL IMPAGLIAZZO, VALENTINE KABANETS, and AVI WIGDERSON. *In search of an easy witness: exponential time vs. probabilistic polynomial time*. J. Comput. Syst. Sci., 65(4):672–694, 2002. (Preliminary version in *16th Computational Complexity Conference*, 2002).

[KT00] JONATHAN KATZ and LUCA TREVISAN. *On the efficiency of local decoding procedures for error-correcting codes*. In *Proc. 32nd ACM Symp. on Theory of Computing (STOC)*, pages 80–86. 2000.

[Lok09] SATYANARAYANA V. LOKAM. *Complexity lower bounds using linear algebra*. Foundations and Trends in Theoretical Computer Science, 4(1-2):1–155, 2009.

[Mie09] THILO MIE. *Short PCPPs verifiable in polylogarithmic time with $o(1)$ queries*. Ann. Math. Artif. Intell., 56(3-4):313–338, 2009.

[MR08] DANA MOSHKOVITZ and RAN RAZ. *Sub-constant error low degree test of almost-linear size*. SIAM J. Comput., 38(1):140–180, 2008. (Preliminary version in *38th STOC*, 2006). eccc:2005/TR05-086.

[Par20] ORR PARADISE. *Smooth and strong PCPs*. In THOMAS VIDICK, ed., *Proc. 11th Innovations in Theor. Comput. Sci. (ITCS)*, volume 151 of *LIPIcs*, pages 2:1–2:41. Schloss Dagstuhl, 2020. eccc:2019/TR19-023.

[Raz89] ALEXANDER A. RAZBOROV. *On rigid matrices (in Russian)*. Technical report, Steklov Mathematical Institute, 1989.

[SSS97] MOHAMMAD AMIN SHOKROLLAHI, DANIEL A. SPIELMAN, and VOLKER STEMANN. *A remark on matrix rigidity*. Inform. Process. Lett., 64(6):283–285, 1997.

[SW14] RAHUL SANTHANAM and R. RYAN WILLIAMS. *On uniformity and circuit lower bounds*. Comput. Complexity, 23(2):177–205, 2014. (Preliminary version in *28th Computational Complexity Conference*, 2013).

[Tre13] LUCA TREVISAN. *Inapproximability of combinatorial optimization problems*. In VANGELIS TH. PASCHOS, ed., *Paradigms of Combinatorial Optimization*, chapter 13, pages 381–434. Wiley Publishers, 2013. arXiv:cs.CC/0409043, eccc:2004/TR04-065.

[Val77] LESLIE G. VALIANT. *Graph-theoretic arguments in low-level complexity*. In JOZEF GRUSKA, ed., *Proc. 6th Symposium of Mathematical Foundations of Computer Science (MFCS)*, volume 53 of *LNCS*, pages 162–176. Springer, 1977.

[Vio20] EMANUELE VIOLA. *New lower bounds for probabilistic degree and AC^0 with parity gates*, 2020. (manuscript). eccc:2020/TR20-015.

[Wil13] R. RYAN WILLIAMS. *Improving exhaustive search implies superpolynomial lower bounds*. SIAM J. Comput., 42(3):1218–1244, 2013. (Preliminary version in *42nd STOC*, 2010).

[Wil14] ———. *Nonuniform ACC circuit lower bounds*. J. ACM, 61(1):2:1–2:32, 2014. (Preliminary version in *Computational Complexity Conference*, 2011).

[Wil16] ———. *Natural proofs versus derandomization*. SIAM J. Comput., 45(2):497–529, 2016. (Preliminary version in *45th STOC*, 2013). arXiv:1212.1891.

[Wun12] HENNING WUNDERLICH. *On a Theorem of Razborov*. Comput. Complexity, 21(3):431–477, 2012. eccc:2010/TR10-086.

[Zák83] STANISLAV ZÁK. *A Turing machine time hierarchy*. Theoret. Comput. Sci., 26:327–333, 1983.