Electronic Colloquium on Computational Complexity, Revision 5 of Report No. 171 (2019)

Improved bounds on the AN-complexity of $O(1)$-linear functions

Oded Goldreich
Department of Computer Science
Weizmann Institute of Science, Rehovot, Israel
E-mail: oded.goldreich@weizmann.ac.il

March 12, 2022

Abstract

We consider arithmetic circuits with arbitrary gates for computing Boolean functions that are represented by *low degree* polynomials over GF(2). An adequate complexity measure for such circuits is the maximum between the arity of the gates and their number. This model and the corresponding complexity measure, called AN-complexity, were introduced by Goldreich and Wigderson (*ECCC*, TR13-043, 2013), and it is meaningful only for low degree polynomials (where the arity of a gate is not due to the degree of the polynomial that the gate computes but rather to the number of variables in it).

The AN-complexity of a function yields an upper bound on the size of depth-three Boolean circuits for computing the function. Specifically, the depth-three size of Boolean circuits is at most exponential in the AN-complexity of the function. Hence, proving linear lower bounds on the AN-complexity of explicit $O(1)$-linear functions is a essential step towards proving that depth-three Boolean circuits for these functions requires exponential size.

In this work, we present explicit $O(1)$-linear functions that require *depth-two* arithmetic circuits of almost linear AN-complexity. Specifically, for every $\epsilon > 0$, we show an explicit $\mathrm{poly}(1/\epsilon)$-linear function $f : \{0,1\}^{\mathrm{poly}(1/\epsilon)\cdot n} \to \{0,1\}$ such that any depth-two arithmetic circuit that computes f must use gates of arity at least $n^{1-\epsilon}$. In particular, for every $\epsilon > 0$ and $t = O(1/\epsilon^2)$, the $\Omega(n^{1-\epsilon})$ lower bound holds also for the t-linear function

$$f(x^{(1)}, x^{(2)}, ..., x^{(t)}) = \sum_{i_1,...,i_{t-1}\in[n]} \left(\prod_{j\in[t-1]} x^{(j)}_{i_j} \right) \cdot x^{(t)}_{i_1+i_2+\cdots+i_{t-1}}$$

This improves over a corresponding lower bound of $\widetilde{\Omega}(n^{2/3})$ that was known for an explicit tri-linear function (Goldreich and Tal, *Computational Complexity*, 2018), but leaves open the problem of showing similar AN-complexity lower bounds for arithmetic circuits of larger depth.

A key aspect in our proof is considering many (extremely skewed) random restrictions, and contrasting the sum of the values of the original function and the circuit (which supposedly computes it) taken over a (carefully chosen) subset of these random restrictions. We show that if the original circuit has too low AN-complexity, then these two sums cannot be equal, which yields a contradiction.

ISSN 1433-8092

Contents

1 Introduction

Providing exponential lower bounds on the size of constant-depth (unbounded fan-in) Boolean circuits that compute explicit functions is a central problem of circuit complexity, even when restricting attention to depth-three circuits (cf., e.g., [5, Chap. 11]). We stress that we refer to lower bounds of the form $\exp(\Omega(n))$, when n is the input length, whereas the celebrated lower bounds on the size of depth-three circuits for parity have the form $\exp(\Omega(n^{1/2}))$.

Focusing on this challenge, Goldreich and Wigderson [4] made two suggestions. The first suggestion was to consider, as potential candidates, explicit Boolean functions that can be represented as low degree polynomials over GF(2), while admitting that it is not even known whether there exist such functions that require exponential size depth-three Boolean circuits. The second suggestion was to start by establishing a size lower bound in a restricted model of depth-three Boolean circuits, which they called *canonical*. The latter model covers the standard construction of depth-three Boolean circuits for parity, and seems quite natural in the context of computing low degree polynomials.

More specifically, for small values of t (e.g., $t = O(1)$ or $t = \mathrm{poly}(\log(n))$), we shall consider functions over $t \cdot n$ variables, partitioned into t blocks, each containing n variables, such that the function is a sum of monomials that contain a single variable from each block (see Eq. (1) coming next). We call such functions t-linear. We shall prove lower bounds on the size of *canonical* depth-three Boolean circuits that compute some explicit $O(1)$-linear functions; that is, we shall prove lower bounds in a restricted but meaningful model of depth-three Boolean circuits.

Since the restricted model of ("canonical") depth-three circuits and its previous studies are not well-known, we start by reviewing this model and these results. Specifically, in Section 1.1 we review the connection between canonical (depth-three) Boolean circuits and a model of arithmetic circuits with general gates of bounded arity. In particular, the size of canonical circuits corresponds to a complexity measure, called *AN-complexity*, of the arithmetic circuits. In Section 1.2 we review the known results regarding the AN-complexity of functions, which lead to open problems that are stated at the beginning of Section 1.3. Our results resolve one of these open problems, and in Section 1.4 we outline our main proof ideas.

We stress that the current work actually studies the AN-complexity of some explicit $O(1)$-linear functions. Hence, the actual setting of this work is the model of arithmetic circuits with general gates and the AN-complexity of such circuits. The connection to (canonical) depth-three Boolean circuits is one (and, in fact, the original) motivation for that study. An additional motivation is provided by the gap between the known lower bounds on (the AN-complexity of) explicit and non-explicit functions.

1.1 From canonical Boolean circuits to AN-complexity of multilinear circuits

As stated above, the original motivation for this work is the study of the size of depth-three Boolean circuits of a restricted type that compute explicit $O(1)$-linear functions. The latter depth-three Boolean circuits, called *canonial*, are obtained by a natural transformation of arithmetic circuits with general gates (i.e., gates that compute arbitrary polynomials). The latter arithmetic circuits are restricted only by the arity of these general gates and their numbers, where these quantitative restrictions serve as the complexity measures of such circuits, called *AN-complexity*.[1] The aforementioned transformation is given in [4, Const. 2.8] and it yields (so-called canonical) Boolean

[1] Indeed, 'A' stands for arity and 'N' stands for number.

circuits of depth three and size that is exponential in the AN-complexity of the arithmetic circuits. Hence, the study of the size of canonical Boolean circuits that compute $O(1)$-linear functions is actually a study of the AN-complexity of these functions.

Additional motivation for the study of the AN-complexity of functions arises from the fact that it is a relatively natural and simple complexity measure that is also plagued by the typical complexity theoretic curse: *the known lower bounds on the complexity of explicit functions fall short of the maximal complexity established by non-explicit functions.*

1.1.1 The functions: t-linearity

As stated above, for small values of t (e.g., $t = O(1)$ or $t = \text{poly}(\log(n))$), we shall consider functions over $t \cdot n$ variables, partitioned into t blocks, each containing n variables, such that the function (viewed as a polynomial over GF(2)) is linear in each of the blocks. Such functions are usually called set-multilinear [8] or block-multilinear, but for simplicity we shall call them multi-linear. Specifically, we say that $f : \{0,1\}^{t \cdot n} \to \{0,1\}$ is t-linear if

$$f(x^{(1)}, x^{(2)}, ..., x^{(t)}) = \sum_{i_1, i_2, ..., i_t \in [n]} f_{i_1,...,i_t} \cdot x^{(1)}_{i_1} \cdot x^{(2)}_{i_2} \cdots x^{(t)}_{i_t}, \tag{1}$$

where $x^{(j)} = (x^{(j)}_1, ..., x^{(j)}_n)$ is the j^{th} block of n Boolean variables.

That is, f is the sum of monomials (or t-way products) such that each monomial takes a single variable from each of the t blocks of variables. The t-ary array $(f_{i_1,...,i_t})_{i_1,...,i_t \in [n]}$ is the tensor that represents the monomials of f; that is, the tensor indicates which of the possible n^t monomials are actually included in the t-linear function. In the special case of $t = 2$ (i.e., bilinear functions), this (two-dimensional) tensor is an n-by-n matrix.

(Jumping ahead, we mention that, as in [4, 3], we shall consider residual bilinear functions obtained by *random restrictions* of explicit $O(1)$-linear functions, and relate the AN-complexity of the original $O(1)$-linear functions to measures of the matrices that represent the residual bilinear functions. In particular, we shall consider the rank of the aforementioned matrices.)

1.1.2 The arithmetic circuits: the multilinearity condition

We consider arithmetic circuits with arbitrary gates that compute multi-linear functions (or rather polynomials). Each gate in such a circuit computes an arbitrary polynomial; the circuit is only restricted by the *arity of the gates* that it uses and their number.

Following the norm in the study of arithmetic circuits, we actually consider the formal polynomial computed by such an arithmetic circuit; that is, the circuit computes an element in the ring of polynomials over the two-element field GF(2). Recall that in this ring $x + x = 0$, but x^2 and x are different polynomials (i.e., different element of the ring). Furthermore, since we consider the computation of multi-linear polynomials, it makes sense to assume that the circuits are multilinear in the sense that each sub-circuit (rooted at any gate) computes a multi-linear function.

Definition 1.1 (multilinear circuits with general gates [4]): *A* multilinear circuit *on t blocks of inputs $x^{(1)}, ..., x^{(t)} \in \{0,1\}^n$ is a directed acyclic graph whose nodes are associated with arbitrary arithmetic gates such that the arithmetic sub-circuit rooted at each gate computes a polynomial over* GF(2) *that is a linear function in the variables of each block on which the polynomial depends; that is, for every gate in a multilinear circuit, there exists a non-empty set $B \subseteq [t]$ such that the*

arithmetic sub-circuit rooted at this gate computes a polynomial that is linear in the variables of each block $j \in B$.

(Indeed, the underlying notion is of B-multilinearity, where an input-gate that is fed by the variable $x_i^{(j)}$ is $\{j\}$-multilinear, and each monomial computed by a B-multilinear (non-input) gate is the product of the results fed by gates that are multilinear in sets that form a partition of B.)[2]

We mention that, when considering the computation of t-linear polynomials, the restriction to multilinear circuits can be imposed by increasing the arity and number of gates in the general arithmetic circuit by at most a factor of 2^t, without increasing the depth of the circuit [4, Rem. 2.5].

1.1.3 The complexity measure: AN-complexity

Clearly, any t-linear function $f : \{0,1\}^{t \cdot n} \to \{0,1\}$ can be computed by a multilinear circuit having a single gate of arity tn. But we are interested in circuits that use gates of *bounded arity*, and we also bound the number of gates that they use.

Definition 1.2 (the AN-complexity of multilinear circuits with general gates [4]): *The* arity *of a multilinear circuit is the maximum arity of its* (general) *gates, and the* number of gates *counts only the general gates and not the leaves* (variables). *The* AN-complexity *of a multilinear circuit is the maximum between its arity and the number of its* (general) *gates.*

- *The* general (or unbounded-depth) AN-complexity *of a multi-linear function f, denoted $\mathtt{AN}(f)$, is the minimum AN-complexity of a multilinear circuit that computes f.*
- *The* depth-two AN-complexity *of a multi-linear function f, denoted $\mathtt{AN}_2(F)$, is the minimum AN-complexity of a* depth-two *multilinear circuit that computes f.*

Indeed, when dealing with depth-two multilinear circuits, there is no need to upper-bound the number of gates, since it is upper-bounded by the arity of the top gate (plus 1).

1.1.4 Obtaining canonical Boolean circuits

A straightforward implementation of general gates of arity m by CNFs (or DNFs) of size $\exp(m)$ yields depth-three circuits of size $\exp(\mathtt{AN}_2(f))$ for any multi-linear function f. (Indeed, we use a CNF for emulating the top multi-linear gate, and DNFs for the intermediate multi-linear gates (and then collapse the two adjacent layers of OR-gates).) The Boolean circuits obtained from this transformation are considered canonical. Note that the upper bound on the size of depth-three Boolean circuits computing `Parity` is obtained by a canonical circuit, which is derived by starting from the depth-two arithmetic circuit that computes an n-way sum by summing up $\sqrt{n}$ sums, each depending on a different set of $\sqrt{n}$ variables.

A more general notion of canonical circuits arises by transforming multilinear circuits of arbitrary depth and bounded AN-complexity. Specifically, applying a Valiant-like idea [10], which can be actually traced to the reduction of Circuit-SAT to SAT, Goldreich and Wigderson [4] showed the following.

[2] That is, for every such monomial, there is a partition, denoted $(B_1, ..., B_w)$, of B such that this monomial is a product of w results and the i^{th} result is fed by a B_i-multilinear gate.

Theorem 1.3 (the size of depth-three Boolean circuits is at most exponential in the AN-complexity [4, Sec. 2.2]): *Any multilinear function f can be computed by a depth-three Boolean circuit of size* $\exp(\mathtt{AN}(f))$.

The Boolean circuits obtained from this transformation are called canonical.

Hence, establishing lower bounds on the AN-complexity of multi-linear functions is a necessary condition for establishing lower bounds on the size of depth-three Boolean circuits for these functions. In particular, seeking lower bounds of the form $\exp(\omega(n^{1/2}))$ on the size of a depth-three Boolean circuit computing the $O(1)$-linear function $f : \{0,1\}^{O(n)} \to \{0,1\}$ requires proving that $\mathtt{AN}(f) = \omega(n^{1/2})$, which in turn requires $\mathtt{AN}_2(f) = \omega(n^{1/2})$.

1.2 Previous results regarding the AN-complexity of multi-linear functions

The following results provide the context for our work. For starters, note that the case of $t = 1$ corresponds to the n-bit parity function $\mathtt{PAR}_n$, and in this case it is easy to verify that $\mathtt{AN}_2(\mathtt{PAR}_n) = O(n^{1/2})$ and $\mathtt{AN}(\mathtt{PAR}_n) = \Omega(n^{1/2})$. Hence, we are interested in the case $t > 1$. Specifically, we seek $O(1)$-linear functions that have AN-complexity $\omega(n^{1/2})$. We start with an upper bound that sets the limit on such lower bounds.

Theorem 1.4 (a generic upper bound [4, Thm. 3.1]): *For every $t \geq 2$, every t-linear function $f : (\{0,1\}^n)^t \to \{0,1\}$ can be computed by depth-two mulitilinear circuit of AN-complexity $O((tn)^{t/(t+1)})$; that is, $\mathtt{AN}_2(f) = O((tn)^{t/(t+1)})$.*

For example, all bilinear functions have AN-complexity at most $O(n^{2/3})$. (We stress that Theorem 1.4 holds also for t that varies with n.) Hence, seeking a linear (in tn) lower bound, we must use a logarithmic number of blocks (i.e., $t = \Omega(\log n)$). In fact, such lower bounds hold existentially.

Theorem 1.5 (existential lower bound [4, Thm. 4.1]): *For every $t \geq 2$, almost all t-linear functions $f : (\{0,1\}^n)^t \to \{0,1\}$ have AN-complexity $\Omega((tn)^{t/(t+1)})$; that is,*

$$\Pr_{f:(\{0,1\}^n)^t \to \{0,1\}}[\mathtt{AN}(f) = \Omega(tn)^{t/(t+1)})] = 1 - o(1).$$

For example, almost all bilinear functions have AN-complexity at least $\Omega(n^{2/3})$. (Again, Theorem 1.5 holds also for t that varies with n.)

Of course, the goal is obtaining lower bounds for explicit functions (and Theorems 1.4 and 1.5 merely set the target for such attempts). The only prior $\omega(n^{1/2})$ result of this type was proved by Goldreich and Tal [3], by building on a connection between the AN-complexity of bilinear functions and matrix rigidity (cf. [9]), which was established by Goldreich and Wigderson [4].

Theorem 1.6 ($\widetilde{\Omega}(n^{1/2})$ lower bounds for explicit functions [3, Thm. 1.5&1.6]):

1. *There exists a polynomial-time computable 4-linear function $f_4 : \{0,1\}^{4n} \to \{0,1\}$ having AN-complexity $\widetilde{\Omega}(n^{2/3})$; that is, $\mathtt{AN}(f_4) = \widetilde{\Omega}(n^{2/3})$.*
2. *The 3-linear function $f_3(x,y,z) = \sum_{i,j \in [n/2]} x_i y_j z_{i+j}$ satisfies $\mathtt{AN}_2(f_3) = \widetilde{\Omega}(n^{2/3})$.*

We mention that the function $f_4(x,y,r,s)$ has the form $\sum_{i,j \in [n/O(1)]} g_{i,j}(r,s) \cdot x_i y_j$, where each $g_{i,j} : \{0,1\}^{2n} \to \{0,1\}$ is a bilinear function that describes a bit in the output of a small bias generator that stretches an $2n$-bit long seed into an $\Omega(n^2)$-bit long sequence (see Definition A.1).

1.3 Our results

The obvious open problems raised by the results reviewed in Section 1.2 are

Open Problem 1.7 ($\omega(n^{2/3})$ lower bounds for explicit functions):

1. *Present an explicit $O(1)$-linear function $f : \{0,1\}^{O(n)} \to \{0,1\}$ having AN-complexity $\omega(n^{2/3})$; that is, $\mathtt{AN}(f) = \omega(n^{2/3})$.*
2. *Present an explicit $O(1)$-linear function $f : \{0,1\}^{O(n)} \to \{0,1\}$ satisfying $\mathtt{AN}_2(f) = \omega(n^{2/3})$.*

We mention that the general AN-complexity of multi-linear functions may be lower than its depth-two AN-complexity (cf. [4, Thm 2.3]). We resolve the second problem by proving the following –

Theorem 1.8 (almost linear lower bounds on $\mathtt{AN}_2$-complexity of explicit functions): *For every $\epsilon > 0$, letting $t = \mathrm{poly}(1/\epsilon)$, there exists a polynomial-time computable t-linear function $f : \{0,1\}^{t \cdot n} \to \{0,1\}$ such that $\mathtt{AN}_2(f) = \Omega(n^{1-\epsilon})$.*

We mention that the lower bound holds also when waiving the requirement that the circuit be multilinear; that is, it holds for general depth-two arithmetic circuits that use general gates. This is the case because general arithmetic circuits of AN-complexity m and depth d that compute a t-linear function can be converted to multilinear circuits of AN-complexity $2^t \cdot m$ and depth d that compute the same function [4, Rem. 2.5].

A somewhat nicer result. Using the ideas that underlie the proof of Theorem 1.8, and triggered by a discussion with Avi Wigderson, we also prove the following result that refers to a more explicit function and a smaller value of t.

Theorem 1.9 (almost linear lower bounds on $\mathtt{AN}_2$-complexity of more explicit functions): *For every $\epsilon \in (0, 1/4]$, letting $t = \lceil 4/\epsilon^2 \rceil$, the t-linear function $f : (\{0,1\}^n)^{t-1} \times \{0,1\}^{(t-1)n} \to \{0,1\}$*

$$f(x^{(1)}, x^{(2)}, ..., x^{(t)}) = \sum_{i_1,...,i_{t-1} \in [n]} \left(\prod_{j \in [t-1]} x^{(j)}_{i_j} \right) \cdot x^{(t)}_{i_1+i_2+\cdots+i_{t-1}}$$

satisfies $\mathtt{AN}_2(f) = \Omega(n^{1-\epsilon})$.

Alternatively, we may partition the last block, which contains $(t-1) \cdot n$ variables, to $t-1$ blocks holding n variables each.[3] We note that this function can be computed in quadratic time (see Remark 3.3).

We mention that the functions used in Theorems 1.8 and 1.9 are generalizations of the 4-linear and trilinear functions used in establishing Part 1 and Part 2 of Theorem 1.6, respectively. Note that all these lower bounds (i.e., Theorem 1.6 as well as Theorems 1.8 and 1.9) fall short of the optimal bounds asserted in Theorems 1.4 and 1.5 (which use $t = \lceil 1/\epsilon \rceil - 1$).

Since the proof of Theorem 1.8 is somewhat simpler than the proof of Theorem 1.9, we only review the former in the introduction. The proof of Theorem 1.9 is quite similar, but involves some additional details (see comment at the end of Section 1.4.4).

[3] As stated, this yields a non-homogenous function; that is, different monomials depend on different subsets of the variable-blocks. To obtain a homogenous function we may augment each variable-block by an auxiliary variable (which will be set to 1) and augment the monomials with these variables.

1.4 Outline of the proof of Theorem 1.8

Let us take a look at a generic multilinear circuit of *depth two* and AN-complexity m that computes a t-linear function f (and recall that our aim is to prove that $m \geq n^{1-\epsilon}$, where $\epsilon = (1/t)^{\Omega(1)}$). This circuit has the form $C(\overline{x}) = H(G_1(\overline{x}), ..., G_m(\overline{x}))$, where me may assume (w.l.o.g.) that the top gate H is only fed by gates (rather than also by variables).[4] Hence, the polynomial computed by this circuit has the form

$$\sum_{J \in \mathcal{C}} \prod_{j \in J} G_j(\overline{x}), \tag{2}$$

where $\mathcal{C}$ is a collection of subsets of $[m]$. Furthermore, each of the intermediate gates (i.e., the G_i's) is fed by variables only; moreover, G_i is fed by at most m variables from each block.[5]

We consider the blocks of variables that feed each of the m intermediate gates, denoting by $B_i \subseteq [t]$ the set of variable-blocks that feed G_i. By the multilinearity of C, each monomial computed by G_i is linear in the variables of each block in B_i and is independent of variables of blocks in $[t] \setminus B_i$. Furthermore, for every $J = \{j_1, ..., j_w\} \in \mathcal{C}$ (as in Eq. (2)), it holds that $(B_{j_1}, ..., B_{j_w})$ is a partition of $[t]$.

It is instructive to distinguish between gates that are fed by many (e.g., more than $2/\epsilon$) blocks and those fed by few blocks. We will handle gates G_i's with large B_i's by using a very sparse random assignment to the first $t-2$ blocks; specifically, in each of these $t-2$ blocks, we assign a single (random) variable the value 1 and set all other variables to 0. Hence, with high probability, each gate G_i with large B_i will evaluate to 0 in the resulting simplified circuit. As shown in [4], the simplified circuit computes a bilinear function that is represented by a matrix of rigidity smaller than m^3 for rank m, and this may (at best) lead to an AN2-complexity lower bound of $\Omega(n^{2/3})$, which was essentially obtained in [3]. To do better, we consider many (i.e., more than m) random restrictions (or assignments) of the foregoing type, and study a carefully chosen linear combination of the corresponding simplified circuits (along with the corresponding slices of f).[6]

1.4.1 The random restriction and its effect on gates with many blocks

Suppose that we assign the variables in the first $t-2$ blocks at random such that for each $j \in [t-2]$ we set a single variable of block j to 1 and set all other variables to 0; that is, for each $j \in [t-2]$, we select $i_j \in [n]$ uniformly at random and set the j^{th} block to $0^{i_j-1}10^{n-i_j}$. Then, the t-linear function $f : \{0,1\}^{t \cdot n} \to \{0,1\}$ is (randomly) restricted to a bilinear function $f_{i_1,...,i_{t-2}} : \{0,1\}^{n+n} \to \{0,1\}$ such that

$$f_{i_1,...,i_{t-2}}(y,z) \stackrel{\text{def}}{=} f(0^{i_1-1}10^{n-i_1}, ..., 0^{i_{t-2}-1}10^{n-i_{t-2}}, y, z) \tag{3}$$

which is supposedly computed by the simplified circuit (i.e., the circuit obtained from C by applying the random restriction specified by $(i_1, ..., i_{t-2})$).

Terminology. Looking at Eq. (1), recall that the monomials of f are *represented* by the t-ary array $(f_{i_1,...,i_t})_{i_1,...,i_t \in [n]}$, and note that $f_{i_1,...,i_t} = f(0^{i_1-1}10^{n-i_1}, ..., 0^{i_t-1}10^{n-i_t})$. Analogously, for

[4] Indeed, for notational simplicity, we consider m intermediate gates, although the AN-complexity upper-bounds their number by $m-1$.

[5] Indeed, G_i is fed by at most m variables (total), but we only use the stated implication (i.e., that it is fed by at most m variables from each block).

[6] We use $d = \Omega(1/\epsilon)$ for handling gates that are fed by more than d blocks, whereas $d = O(1/\epsilon)$ is used for handling the other gates (where we rely on $n^{d/(t-2)} \ll n^{\epsilon}$, which implies $t = \Omega(d/\epsilon)$).

each $(i_1, ..., i_{t-2}) \in [n]^{t-2}$, the n-by-n matrix $(f_{i_1,...,i_{t-2},i_{t-1},i_t})_{i_{t-1},i_t \in [n]}$ represents the (monomials of the) bilinear function $f_{i_1,...,i_{t-2}} : \{0,1\}^{n+n} \to \{0,1\}$, and this matrix may be viewed as a slice of the t-ary tensor that represents the monomials of f. Specifically, the matrix $(f_{i_1,...,i_t})_{i_{t-1},i_t \in [n]}$ is obtained from the t-ary array $(f_{i_1,...,i_t})_{i_1,...,i_t \in [n]}$ by fixing the first $t-2$ coordinates of the t-ary array such that, for each $j \in [t-2]$, the j^{th} coordinate is fixed to i_j. Indeed, $f_{i_1,...,i_t} = f_{i_1,...,i_{t-2}}(0^{i_{t-1}-1}10^{i_{t-1}}, 0^{i_t-1}10^{i_t})$, which equals $f(0^{i_1-1}10^{n-i_1}, ..., 0^{i_t-1}10^{n-i_t})$. Hence, we may also view the bilinear functin $f_{i_1,...,i_{t-2}}$ as a slice of the t-linear function f.

We now look at the effect of the foregoing random restriction on the circuit C, which supposedly computes f. Looking at the simplified circuit, observe that, for each $i \in [m]$, the output of G_i is identically 0 unless, for each $j \in B_i$, we set one of the variables of block j that feeds G_i to 1 (i.e., if i_j is the index of one of the variables of block j that feeds G_i). Recalling that each block has at most m variables that feed G_i, it follows that the output of G_i is identically 0 with probability at least $1 - (m/n)^{|B_i \cap [t-2]|}$.

Now, if all B_i's were of size at least $d+2 > 1/\epsilon$, then the entire circuit would simplify to the constant 0, with probability at least $1 - m \cdot (m/n)^d$. Hence, if all slices of f are non-trivial (i.e., none of $f_{i_1,...,i_{t-2}}$'s is identically 0), then we reach a contradiction unless $m \cdot (m/n)^d \geq 1$, which implies $m \geq n^{d/(d+1)} > n^{1-\epsilon}$, and Theorem 1.8 would follow (when we pick the best depth-two circuit; i.e., $m = \mathtt{AN}_2(f)$).

1.4.2 Gates with few blocks: A special case

But what if some (or all) B_i's are small? For simplicity, let us ignore the large B_i's, and more importantly assume that each of the remaining B_i's is either contained in $\{t-1, t\}$ or in $[t-2]$ (i.e., if G_i is fed by some block in $\{t-1, t\}$, then it is not fed by any block in $[t-2]$). That is, here we consider only a special case, which suffices for introducing one key idea. The general case, where we handle all gates G_i that have a small B_i, is postponed to Section 1.4.3. Now, in the special case (i.e., either $B_i \subseteq \{t-1, t\}$ or $B_i \subseteq [t-2]$), the circuit C has the form

$$C(x^{(1)}, ..., x^{(t-2)}, y, z) = \sum_{j \in J} F_j(x^{(1)}, ..., x^{(t-2)}) \cdot G_j(y, z) \quad (4)$$

$$+ \sum_{(j_1,j_2) \in K} F_{j_1,j_2}(x^{(1)}, ..., x^{(t-2)}) \cdot G_{j_1}(y) \cdot G_{j_2}(z), \quad (5)$$

where the F_j's and F_{j_1,j_2}'s are $(t-2)$-linear functions, $J \subseteq [m]$ and $K \subseteq \{(j_1, j_2) \in ([m] \setminus J)^2 : j_1 < j_2\}$. (The foregoing form is obtained by recalling that C is a mulitilinear function of the various G_j's, considering only the G_j's that satisfy $B_j \subseteq \{t-1, t\}$, and letting F_j be the factor that multiplies a bilinear G_j, and F_{j_1,j_2} be the factor that multiplies $G_{j_1} \cdot G_{j_2}$.)

Recalling that, in order to get rid of the larger B_i's, we picked a random assignment to the variables in the blocks in $[t-2]$, it follows that each of the functions F_j's and F_{j_1,j_2}'s evaluates to a constant. Now, if we are extremely lucky and all F_j's (but not necessarily the F_{j_1,j_2}'s)[7] evaluate to 0, then the bilinear function computed by the residual circuit simplifies to Eq. (5); that is, this bilinear function equals $\sum_{j_1 \in [t] \setminus J} G_{j_1}(y) \cdot \sum_{j_2:(j_1,j_2) \in K} G_{j_2}(z)$. One key observation (of [4]) is that the matrix that represents the (monomials of the) latter bilinear function has rank at most m,

[7] Footnote 8 explains why we do not assume here that all F_{j_1,j_2} evaluate to 0. Essentially, removing the unrealistic assumption regarding the F_j's has a cost we can afford, whereas the cost of dealing similarly with the F_{j_1,j_2}'s cannot be afforded (because their number may be much larger).

because each $G_{j_1}(y) \cdot \sum_{j_2:(j_1,j_2)\in K} G_{j_2}(z)$ is represented by a matrix of rank at most 1. In this case we reach a contradiction, provided that almost all slices of f correspond to matrices of higher rank (i.e., rank higher than m). Hence, Theorem 1.8 would follow, provided that we select an adequate function f (which is quite easy to do).

Of course, there is no reason to believe that we may be so extremely lucky (i.e., have all F_j's evaluate to 0). What we do instead is select $m+1$ random assignments to the variables of the blocks in $[t-2]$, and take a suitable linear combinations of the $m+1$ vectors that describes the values of the F_j's under these assignments; that is, the i^{th} vector describes the values of all F_j's under the i^{th} assignment. Specifically, we take a non-zero linear combination of these vectors that sums-up to zero, and note that the corresponding linear combination of the simplied bilinear circuits (obtained from C) is represented by a matrix of rank at most m (see details below). In contrast, the corresponding linear combination of the residual bilinear functions derived from f (by these random restrictions) will be shown to yield a matrix of high rank (i.e., rank greater m).[8]

Let us repeat and detail the argument. The key observation is that the foregoing linear combination of the computations of (the restricted) circuit C yields a bilinear function that is represented by a matrix of rank at most m. This is the case because the contribution of $\sum_{j\in J} F_j(\cdots)G_j(y,z)$ cancels out (since, for each $j \in J$, the linear combination of the values of F_j under these assignments sums-up to 0, whereas G_j is not affected by any of these assignments), and so we are left with Eq. (5). Hence, if the corresponding linear combination of the (restricted) function f yields a bilinear function that is represented by a matrix of higher rank (i..e, higher than m), then we reach a contradiction (as above). For this to happen, it suffices that *each linear combinations of slices of f yields a matrix of rank higher than m*, where a slice of f is the bilinear function $f_{i_1,...,i_{t-2}}$ defined in Eq. (3). Selecting f from a small-bias sample space comes to mind, and such a selection will be derandomized by using auxiliary variables (as done in the construction of f_4 of [3]). Furthermore, as in [3], we need a generator of such sequences (with larger stretch than in [3]) that can be implemented by multi-linear functions of low degree. We present such a construction in the appendix.

But wait: We have ignored the effect of using $m+1$ random assignments, rather than one, on the large B_i's (i.e., $|B_i| \geq d+2$). Recall that when using a single random assignment of the foregoing type, the contribution of the corresponding gate vanished with probability at least $1-(m/n)^{|B_i\cap[t-2]|}$. But when selecting $m+1$ such assignments the corresponding gate vanishes on all of them with probability at least $1-((m+1)\cdot(m/n))^{|B_i\cap[t-2]|}$. This bound is useless, since we aim at $m \gg \sqrt{n}$. However, we can do better by selecting the $m+1$ random assignments carefully. Specifically, for each $j \in [t-2]$, we select a set I_j of $b=(m+1)^{1/(t-2)}$ elements of $[n]$ uniformly at random, and consider the set of assignments specified by $I_1 \times \cdots \times I_{t-2}$. Observe that the i^{th} gate vanishes on all $m+1$ assignments if for some $j \in B_i \cap [t-2]$ it holds that I_j contains no index of a variable of block j that feeds this gate, whereas this event occurs with probability at least $1-(b\cdot(m/n))^{|B_i\cap[t-2]|}$. Hence, analogously to Section 1.4.1, we reach a contradiction unless $m\cdot((b\cdot m)/n)^d \geq 1$. Recalling that $b=(m+1)^{1/(t-2)}$, we infer that $m\cdot((m+1)^{(t-1)/(t-2)}/n)^d > 1$,

[8] Indeed, we shall use the hypothesis that the linear combinations of the residual bilinear functions derived from f (by these random restrictions) yield (n-by-n) matrices that are all of high rank (i.e., rank at least m). This hypothesis will be established by relying on a union bound that can support at most $2^{O(n)}$ different linear combinations, whereas we aim at $m=\omega(n^{1/2})$. Hence, we can afford to consider all $2^{m+1}-1$ non-zero linear combinations of $m+1$ residual bilinear functions, but we can not afford to consider $2^{\Omega(m^2)}$ linear combination of $\Omega(m^2)$ residual bilinear functions. This is the reason that this argument is applied to the F_j's, but not to the F_{j_1,j_2}'s.

which implies that $m+1 > n^{\frac{(t-2)d}{(t-1)(d+1)-1}} > n^{1-\epsilon}$, provided that $\min(d+1, t-2) \geq 2/\epsilon$.

1.4.3 Gates with few blocks: The general case

All the foregoing was done under the (unjustified) simplifying assumption that each of the small B_i's is either a subset of $\{t-1,t\}$ or a subset of $[t-2]$. In the general case, we may have gates with small B_i's that intersect both $\{t-1,t\}$ and $[t-2]$.

We first consider the case of small B_i's that contain both $t-1$ and t (i.e., $B_i \supseteq \{t-1,t\}$). A generic gate G_i of this type depends on less than d blocks from $[t-2]$; let us assume for simplicity that these blocks are indexed $1,...,d'$ (and recall that $d' < d$). Then, the contribution of G_i to the computation of C has the form

$$F_i(x^{(1)},...,x^{(t-2)}) \cdot G_i(x^{(1)},...,x^{(d')},y,z), \tag{6}$$

where F_i is an arbitrary $(t-2)$-linear function (analogously to Eq. (4), except that G_i also depends on $x^{(1)},...,x^{(d')}$).[9] Observe that fixing the values of $x^{(1)},...,x^{(d')}$ simplifies Eq. (6) to a form that is identical to Eq. (4). This means that for the sets $I_1,...,I_{t-2}$ selected as in Section 1.4.2, it suffices to find a set $S \subseteq I_1 \times \cdots \times I_{t-2}$ such that for every gate G_i of the current form and every $(i_1,...,i_{d'}) \in I_1 \times \cdots \times I_{d'}$, it holds that

$$\sum_{(i_{d'+1},...,i_{t-2}):(i_1,...,i_{d'},i_{d'+1},...,i_{t-2})\in S} F_i(0^{i_1-1}10^{n-i_1},...,0^{i_{t-2}-1}10^{n-i_{t-2}}) = 0.$$

We can find such a set S, very much as we have done before, except that now we need the size of each I_j (denoted b above) to be at least $(m+1)^{1/(t-2-d')}$ (rather than at least $(m+1)^{1/(t-2)}$), because we need the number of non-zero linear combinations (i.e., $b^{t'-2}-1$) to exceed the number of the foregiong conditions (i.e., $b^{d'} \cdot m$). This means that we get $m \cdot ((m+1)^{1/(t-2-d')} \cdot m)/n)^{d'} \geq 1$, which gives us the desired lower bound (when using a larger t such that $b^d \cdot m < n/2$).[10]

So we are left with the case of small B_i's that intersect $\{t-1,t\}$ at a single point, where $d' \stackrel{\text{def}}{=} |B_i \cap [t-2]| < d$. The crucial observation here is that the probability that G_i does not vanish under a random assignment of the foregoing type is at most $(m/n)^{d'}$, and that this event depends only on the variables in the blocks in $B_i \cap [t-2]$. Taking a union bound over all d'-tuples in $I_1 \times \cdots \times I_{d'}$, it follows that the probability that G_i does not vanish under some of the b^{t-2} random assignment (specified by $I_1 \times \cdots \times I_{t-2}$) is at most

$$b^{d'} \cdot (m/n)^{d'} = (m+1)^{d'/(t-2-d)} \cdot (m/n)^{d'} < \left(\frac{(m+1)^{1+\frac{1}{t-2-d}}}{n}\right)^{d'} \tag{7}$$

Recalling that we aim to derive a contradiction to the hypothesis that $m \leq n^{\frac{t-2-d}{t-1-d}}$, we observe that Eq. (7) is maximized in the case $d' = 0$. Hence, the contribution of such a gate to the rank of the matrix that represents the bilinear function computed by the residual circuit is actually

[9] Actually, F_i does not depend on the first d' blocks, but we do not use this fact.

[10] The added condition is used in the analysis of the pseudorandom function f, which is omitted here. Note that, using $d = 2/\epsilon$ and $t \geq \max(2d+2,(d+1)^2)$, it holds that $m^{1+\frac{d}{t-2-d}+d} > n^d$ implies $m > n^{1-\epsilon}$, whereas $b^d = (m+1)^{d/(t-d-1)} < n/2m$ holds for $m = n^{1-\epsilon}$.

maximized in the case $d' = 0$, which was considered in Section 1.4.2 (i.e., the special case (of the cases considered here)).

This completes the rough sketch of the proof of Theorem 1.8, although a more clear and detailed description is in place. In particular, we ignored the task of showing that for a pseudorandom (i.e., small-bias) function f, with high probability, any non-zero linear combination of the slices in $I_1 \times \cdots \times I_{t-2}$ yields a bilinear function that is represented by a matrix of high rank.

1.4.4 Digest

To summarize, we started with an extremely skewed type of random restrictions, which assign values to all but two of the variable blocks such that a single variable in each block is set to 1. Hence, such random restrictions correspond to selecting, at random, a single variable in each of the $t-2$ blocks. Furthermore, we considered b^{t-2} such random assignments and contrasted the sum of the corresponding restrictions of the original function f and of the circuit C (which supposedly computes it), where the sum is taken over a subset of these assignments. Lastly, the b^{t-2} random assignments are the Cartesian product of b assignments (of random n-bit strings of Hamming weight 1) to each of the $t-2$ blocks.

As mentioned briefly, the function f is a pseudorandom function, and we shall use a function F that uses a small bias generator to specify a function f (see Eq. (8), coming next). Specifically, we shall use an $\exp(-\Omega(n))$-bias generator that stretches a $\mathrm{poly}(t) \cdot n$-bit long seed into a sequence of n^t bits (which specifies a t-linear function) such that each output bit can be computed by an explicit $\mathrm{poly}(t)$-linear function. (Recall that the small-bias feature means that each non-zero linear combination of the output bits is $\exp(-\Omega(n))$-close to being unbiased.) An adequate construction is presented in the appendix.

On the proof of Theorem 1.9. While Theorem 1.9 refers to a different function than Theorem 1.8, the analysis of the multilinear depth-two circuit that supposedly computes this function is almost identical, where the only difference is in an auxiliary condition that is imposed on the set S (which specifies the linear combination used in Section 1.4.3). The main difference is in arguing that a linear combination of slices of the corresponding tensor has high rank. In the case of Theorem 1.8 this follows from the pseudorandomness of the tensor (see previous paragraph), whereas in the proof of Theorem 1.9 we shall show that the relevant matrix is a random Toeplitz matrix.

1.5 Organization

The core of this paper is the proof of Theorem 1.8, which is presented in Section 2. Next, in Section 3, we adapt this proof in order to establish Theorem 1.9. Although the function that underlies Theorem 1.9 is simpler than the one used in the proof of Theorem 1.8, we believe that the proof of Theorem 1.9 is slightly more complicated. In Section 4 we extend the main results to any $\epsilon = \omega(1/\sqrt{\log n})$ and discuss some concrete challenges.

2 Proof of Theorem 1.8

We start with an explicit presentation of the multi-linear functions that we shall analyze. The presentation will use slightly different notations than those used in Section 1.4. In particular, in this presentation, we explicitly include the pseudorandom generator that defines a pseudorandom

function f, rather than push it under the rug (as done in Section 1.4). Specifically, we shall present $(t'+t'')$-linear functions, where the first $t'+2$ blocks of variables correspond to the t blocks in Section 1.4, the random restriction will be applied to the first t' blocks, the residual bilinear function that we shall analyze will depend on the variables of blocks $t'+1$ and $t'+2$, and the seed of the pseudorandom generator will occupy the last $t''-2$ blocks.[11]

For t' and t'' to be determined, we consider the $(t'+t'')$-linear function $F : \mathrm{GF}(2)^{(t'+t'')\cdot n} \to \mathrm{GF}(2)$ in which the first $t'+2$ coordinates of the tensor that represents F correspond to a pseudorandom (i.e., small-bias) tensor, where the pseudorandomness is provided by the last $t''-2$ coordinates. Specifically, we use an $\exp(-\Omega(n))$-bias generator, $G_{\mathtt{sb}} : \mathrm{GF}(2)^{(t''-2)\cdot n} \to \mathrm{GF}(2)^{n^{t'+2}}$, that is computed by $(t''-2)$-linear functions; that is, the $(i_1,...,i_{t'+2})^{\text{th}}$ bit in the output of $G_{\mathtt{sb}}$, denoted $G_{\mathtt{sb}}(x^{(t'+3)},...,x^{(t'+t'')})_{(i_1,...,i_{t'+2})}$, is computed by a $(t''-2)$-linear function. Hence, the function F is defined by

$$F(x^{(1)},x^{(2)},...,x^{(t'+t'')}) = \sum_{i_1,...,i_{t'+2}\in[n]} G_{\mathtt{sb}}(x^{(t'+3)},...,x^{(t'+t'')})_{(i_1,...,i_{t'+2})} \cdot \prod_{j\in[t'+2]} x^{(j)}_{i_j} \tag{8}$$

Fixing an arbitrary assignment $s \in \{0,1\}^{(t''-2)n}$ to the last $t''-2$ blocks of variables of F (i.e., fixing a seed to the small-bias generator), we consider the resulting $(t'+2)$-dimensional tensor, which represents a pseudorandom function $F_s(\cdots) = F(\cdots,s)$. Indeed, such a $(t'+2)$-linear function $F_s : \{0,1\}^{(t'+2)n} \to \{0,1\}$ corresponds to the function f in Eq. (1); in particular, the bit $G_{\mathtt{sb}}(s)_{(i_1,...,i_{t'+2})}$ correspond to the coefficient $f_{i_1,...,i_t}$ in Eq. (1), where $t = t'+2$.

We shall consider the $n^{t'}$ bilinear functions that are obtained by restricting F_s by assignments of unit vectors to its first t' blocks. Each of these bilinear functions will be represented by an n-by-n matrix that is a (two-dimensional) slice of the $(t'+2)$-dimensional tensor $(G_{\mathtt{sb}}(s)_{(i_1,...,i_{t'+2})})_{i_1,...,i_{t'+2}\in[n]}$, where the slices are aligned with coordinates $t'+1$ and $t'+2$. Specifically, the slice $(i_1,...,i_{t'}) \in [n]^{t'}$ of the (tensor of the) residual function F_s is the n-by-n matrix $(G_{\mathtt{sb}}(s)_{(i_1,...,i_{t'},j_1,j_2)})_{j_1,j_2}$; that is, the $(j_1,j_2)^{\text{th}}$ entry of this slice is $G_{\mathtt{sb}}(s)_{(i_1,...,i_{t'},j_1,j_2)}$. Recall that, in general, a bilinear function $B : \{0,1\}^{n+n} \to \{0,1\}$ is represented by an n-by-n matrix such that the $(i,j)^{\text{th}}$ entry in the matrix equals $B(0^{i-1}10^{n-i}, 0^{j-1}10^{n-j})$; that is, this entry is the coefficient of the monomial y_iz_j in the polynomial $B(y,z)$.

In order to prove a lower bound of $\Omega(n^{1-\epsilon})$, we shall pick a large enough $t' = \Omega(1/\epsilon)$, and this will require setting $t''-2 = \Omega(t')$ so that the $n^{t'+2}$-long sample space can have small bias. As in Section 1.4, most of our analysis (i.e., Sections 2.2 and 2.3) will refer to a generic $(t'+2)$-linear function, which may be viewed as one of the possible residual function F_s. We shall return the specific function F of Eq. (8) in Section 2.4.

2.1 The form of a depth-two multilinear circuit that computes F

Without loss of generality, the top gate in a generic depth-two circuit of AN-complexity m (which computes F) sums-up monomials that are products of up to $t = t'+t''$ of the m auxiliary functions, denoted $G_1,...,G_m$. The G_i's are computed by the m intermediate gates, which are each fed by m original variables. A typical gate is associated with a subset $B \subset [m]$ of size at most t, which specify the variable-blocks that feed it; specifically, for each $i \in [m]$, we denote by $\mathtt{BL}(i) \subseteq [t]$ the

[11] This specific choice of parameters is used because it will be more convenient to have a parameter (i.e., t') that equals the number of blocks that are randomly restricted.

indices of the variable-blocks on which G_i depends, and we say that G_i is $\mathtt{BL}(i)$-multilinear. The multi-linearity condition implies that each monomial computed by the top gate induces a partition on the t blocks of variables; that is, for a monomial of the form $\prod_{j\in[w]} G_{i_j}$ computed by the top gate it must hold that $\mathtt{BL}(\{i_1,...,i_w\}) \stackrel{\text{def}}{=} (\mathtt{BL}(i_1),...,\mathtt{BL}(i_w))$ is a partition of $[t]$.

Letting Π denote the set of all partitions of $[t]$ and $\mathcal{C} \subseteq \bigcup_{w\in[t]} \binom{[m]}{w}$ denote the collection of all legal monomials (i.e., $\mathcal{C} = \{I : \mathtt{BL}(I) \in \Pi\}$), the fact that a depth-two circuit of AN-complexity m computes F means that there exist constants $(c_I)_{I\in\mathcal{C}}$ such that

$$F = \sum_{I\in\mathcal{C}} c_I \cdot \prod_{i\in I} G_i \tag{9}$$

$$= \sum_{i\in[m]:\mathtt{BL}(i)\ni t'+1} G_i \cdot \sum_{I\in\mathcal{C}:I\ni i} c_I \cdot \prod_{j\in(I\setminus\{i\})} G_j, \tag{10}$$

where Eq. (10) groups the monomials of Eq. (9) according to the function G_i that is fed by variables of block $t'+1$ (where the choice of using block $t'+1$ rather than block $t'+2$ is arbitrary). Recall that each G_i has arity at most m (and an empty product is defined as identical to 1 (i.e., $\prod_{j\in\emptyset} G_j = 1$)). Letting

$$F_i \stackrel{\text{def}}{=} \sum_{I\in\mathcal{C}:I\ni i} c_I \cdot \prod_{j\in(I\setminus\{i\})} G_j, \tag{11}$$

we write Eq. (10) as

$$F = \sum_{i\in[m]:\mathtt{BL}(i)\ni t'+1} G_i \cdot F_i \tag{12}$$

where we treat F_i as an arbitrary $([t]\setminus\mathtt{BL}(i))$-multilinear function. That is, while each G_i depends on at most m variables (which reside in blocks in the corresponding $\mathtt{BL}(i)$), no such restriction is placed on the F_i's (which arise from Eq. (10)).

Note that the r.h.s. of Eq. (12) depends on the (depth-two) circuit that supposedly computes F, whereas the l.h.s. of Eq. (12) is oblivious of that circuit. We shall show that the equality between the two sides of Eq. (12) cannot possibly hold if m is too small (i.e., if $m = o(n^{1-\epsilon})$, where $\epsilon = \sqrt{O(1)/t'}$).

2.2 Three types of gates and how they will be handled

For each fixing of values to the last $t''-2$ blocks (i.e., a fixing of a seed for the small-bias generator), we show that adequate random assignments to the first t' blocks yield, with high probability, a bilinear function that is represented by a matrix of rank $O(m)$, where this bilinear function is derived from the r.h.s. of Eq. (12). In contrast, considering the l.h.s. of Eq. (12), we shall later show that this is unlikely to happen for a random seed (i.e., F_s is likely be represented by a matrix of higher rank), unless $m = \Omega(n^{1-\epsilon})$.

We start by analyzing the r.h.s. of Eq. (12), following the outline provided in Section 1.4 (but using a slightly different organization).[12] Towards this end, we fix such a seed $s \in \{0,1\}^{(t''-2)\cdot n}$,

[12] While in Section 1.4 the main distinction was between large and small B_i's, here the main distinction is between $B_i \ni t'+2$ and $B_i \not\ni t'+2$, where $B_i \ni t'+1$ always holds (by Eq. (12)). Next, for $B_i \supseteq \{t'+1, t'+2\}$, we distinguish between large and small B_i's. The special case considered in Section 1.4.2 does not appear in the high level organization here.

and for sake of simplicity omit it from the notation (i.e., we shall refer to the functions F, G_i, F_i as if they only depend on the first $t'+2$ blocks). Fixing a sufficiently large constant $d = O(1/\epsilon)$, we consider a partition of the G_i's in Eq. (12) to three types:

1. Gates G_i that are not fed by variables from block $t'+2$ (but are fed by variables from block $t'+1$); that is, $t'+2 \notin \mathtt{BL}(i)$ (although $t'+1 \in \mathtt{BL}(i)$). This means that the variables of block $t'+2$ feed the corresponding function F_i (of Eq. (12)).

   Each such gate G_i contributes to the rank of a matrix that represents the bilinear function that is obtained from the r.h.s. of Eq. (12) by hitting the first t' blocks with a random restriction (which assigns each block a uniformly distributed unit vector). This contribution will be bounded by taking into account the number of blocks in $[t']$ that feed G_i. Specifically, if $\mathtt{BL}(i) \cap [t'] = \emptyset$, then G_i contributes at most one unit to the rank, and otherwise its expected contribution to the rank (taken over a random restriction) vanishes exponentially with $|\mathtt{BL}(i) \cap [t']|$ (see Section 1.4.3).

2. Gates G_i that are fed by variables from block $t'+2$ (as well as by variables from block $t'+1$) along with variables from *many* blocks in $[t']$; that is, $t'+1, t'+2 \in \mathtt{BL}(i)$ and $|\mathtt{BL}(i) \cap [t']| \geq d$.

   We shall show that, with high probability (over the choice of the random restriction), the contribution of these gates vanishes; that is, these gates do not contribute to the rank of a matrix that represents the bilinear function that is obtained from the r.h.s. of Eq. (12) by hitting the first t' blocks with a random restriction.

3. Gates G_i that are fed by variables from block $t'+2$ (as well as by variables from block $t'+1$) along with variables from *few* blocks in $[t']$; that is, $t'+1, t'+2 \in \mathtt{BL}(i)$ and $|\mathtt{BL}(i) \cap [t']| < d$.

   We shall show that that these gates do not contribute to a *suitable linear combination* of bilinear functions obtained by random restrictions as above, where the number of linear combinations is kept small enough by relying on the hypothesis $|\mathtt{BL}(i) \cap [t']| < d$. Consequently, these gates do not contribute to the rank of the matrix that represent the resulting bilinear function.

   The fact that we consider several random restrictions rather than a single one will complicate the analysis of the first two cases, but not in an unmanageable manner. In particular, as discussed in Section 1.4, the different random restrictions will be related so that their effect on the first two cases is relatively small.

Combining the analyses of the three types, we shall show that, with probability at least $2/3$, the rank of the matrix that represents the aforementioned bilinear function (i.e., a suitable linear combination of bilinear functions obtained by several random restrictions) is $O(m)$, unless $m = \Omega(n^{1-\epsilon})$. The lower bound on m will follow by showing that, with very high probability, the matrix obtained by a corresponding linear combination of slices of the tensor that represents F itself has rank $\Omega(n)$.

2.3 The actual handling of the three types

Fixing an arbitrary i such that $\mathtt{BL}(i) \ni t'+1$, we now consider in greater detail what happens in each of the foregoing cases, when arbitrarily fixing of the values of the variables in blocks $[t'+3, t'+t'']$. (For notational simplicity, we shall ignore this fixing in the following discussion; that is, we shall consider G_i and F_i as if they only depend on variables that reside in blocks in $[t'+2]$.)

Parameters that will be used: Recall that m is the AN-complexity of the circuit C, which supposedly computes F, and that our aim is to prove that $m = \Omega(n^{1-\epsilon})$ for some predetermined $\epsilon > 0$. Recall that t' and t'' will be determined as a function of ϵ. The same holds for d, which is the threshold that distinguishes Type 2 from Type 3 gates, and b, which specifies the number of random assignments we consider for each block in $[t']$. We shall assume that these parameters satisfy $b \cdot m \leq n$ and $b^{t'-d} > m$ (whereas $b^{t'} < (n/2) - 2$ will be assumed in Section 2.4).

2.3.1 Gates of Type 1: $t' + 2 \notin \mathtt{BL}(i)$

For any fixing of the values of the variables in all blocks in $[t']$, the function $F_i \cdot G_i$ simplifies to a bilinear function in $x^{(t'+2)}$ and $x^{(t'+1)}$; that is, F_i simplifies to a linear function that depends only on $x^{(t'+2)}$ whereas G_i simplifies to a linear function depends only on $x^{(t'+1)}$. Consequently, *the n-by-n matrix that represents this residual bilinear function has rank at most 1*, because it is an outer product of two vectors (i.e., the vectors representing the simplified F_i and G_i). Letting T_1 denote the set of all i's of Type 1, we note that under the foregoing fixing of values (to all variables in blocks in $[t']$) the n-by-n matrix that represents the residual bilinear function resulting from $\sum_{i \in T_1} F_i G_i$ has rank at most $|T_1| \leq m$.

Foreseeing the treatment of Type 3, which was outlined in Section 1.4.3, we need to handle the sum of $F_i \cdot G_i$ taken over $b^{t'}$ random assignments to the variables (in the blocks in $[t']$). The problem is that each such random assignment may yield a different bilinear function in $x^{(t'+1)}$ and $x^{(t'+2)}$, which is represented by a different matrix of rank at most 1. Our aim is to show that even in this case, the expected rank of the sum of these matrices is at most 1. Intuitively, this is the case because each of the foregoing bilinear functions vanishes with probability at least $1 - b^{-|\mathtt{BL}(i) \cap [t']|}$, where the probability is taken over the choice of the $b^{t'}$ random assignments.[13] Actually, we show that G_i itself vanishes with such probability, and note that this event depends only of the assignment to the blocks in $\mathtt{BL}(i) \cap [t']$. Using a specific set of $b^{t'}$ (related) random assignments, which is the Cartesian product of b random assignments to each of the t' blocks, the expected number of non-vanishing bilinear functions is at most 1 (just as in the simple case that $\mathtt{BL}(i) \cap [t'] = \emptyset$). Details follow.

As stated above, our analysis relies on the specific set of $b^{t'}$ related random assignments that we use. Each of these random assignments is a sequence of t' random unit vectors (i.e., the sequence of n variables of block $j \in [t']$ is assigned the unit vector $0^{i_j - 1}10^{n-i_j}$, where i_j is uniformly distributed in $[n]$), but the different random assignments are related. Specifically, the $b^{t'}$ assignments are specified by a random sequence of b-subsets, denoted $\overline{I} = (I_1, ..., I_{t'})$, such that for every $(i_1, ..., i_{t'}) \in I_1 \times \cdots \times I_{t'}$, we consider the assignment $(\mathtt{u}(i_1), ..., \mathtt{u}(i_{t'})) \in \{0,1\}^{t' \cdot n}$, where $\mathtt{u}(i) = 0^{i-1}10^{n-i}$ is the i^{th} unit vector. We index each of these $b^{t'}$ random assignments with an t'-long sequence over $[b]$. Specifically, for each $j \in [t']$ and $k \in [b]$, we denote by $I_j(k)$ the k^{th} element in I_j. Hence, each $(k_1, ..., k_{t'}) \in [b]^{t'}$ specifies one of the chosen random sequence $(I_1(k_1), ..., I_{t'}(k_{t'})) \in [n]^{t'}$, which in turn specifies a random assignment $(\mathtt{u}(I_1(k_1)), ..., \mathtt{u}(I_{t'}(k_{t'}))) \in \{0,1\}^{t' \cdot n}$.

Digest of the new notations: Recall that each random assignment that we consider (to the t' first blocks) is a sequence of t' random unit vectors, where each unit vector is specified by an element of $[n]$. We shall consider a set of $b^{t'}$ random assignments that are specified by a t'-long sequence of b-subsets of $[n]$, where $b^{t'} \gg m$ (but $b^{t'} < n/2$).

[13] This probability bound relies on $m/n \leq 1/b$.

1. For $I \in \binom{[n]}{b}$ and $k \in [b]$, we let $I(k)$ denote the k^{th} element in I, where the order of elements in I is arbitrary.

   Actually, for concreteness and simplicity, we assume that the ordering is random; that is, for every $k \in [b]$, and a uniformly distributed $I \in \binom{[n]}{b}$ it holds that $I(k)$ is uniformly distributed in $[n]$.

2. For $\overline{I} = (I_1, ..., I_{t'}) \in \binom{[n]}{b}^{t'}$, and $(k_1, ..., k_{t'}) \in [b]^{t'}$, it holds that $(I_1(k_1), ..., I_{t'}(k_{t'})) \in [n]^{t'}$.

3. For $i \in [n]$, the i^{th} unit vector is $\mathtt{u}(i) = 0^{i-1}10^{n-i}$.

   Hence, for $\overline{I} = (I_1, ..., I_{t'}) \in \binom{[n]}{b}^{t'}$, and $(k_1, ..., k_{t'}) \in [b]^{t'}$, it holds that $(\mathtt{u}(I_1(k_1)), ..., \mathtt{u}(I_{t'}(k_{t'})))$ is a sequence of t' unit vectors such that the j^{th} vector in this sequence is $0^{I_j(k_j)-1}10^{n-I_j(k_j)}$.

Indeed, $\mathtt{u}(I_j(k_j)) = 0^{i-1}10^{n-i}$ if the k_j^{th} element in I_j eqauls i.

We wish to upper-bound, for a random sequence $\overline{I}$, and any $S = S(\overline{I}) \subseteq [b]^{t'}$ which may depend on $\overline{I}$, the rank of the matrix that represents the bilinear function

$$\sum_{(k_1,...,k_{t'}) \in S} F_i(\mathtt{u}(I_1(k_1)), ..., \mathtt{u}(I_{t'}(k_{t'})), x^{(t'+2)}) \cdot G_i(\mathtt{u}(I_1(k_1)), ..., \mathtt{u}(I_{t'}(k_{t'})), x^{(t'+1)}), \tag{13}$$

where we omitted $x^{(t'+1)}$ (resp., $x^{(t'+2)}$) from F_i (resp., from G_i) because this block is not fed to that function. We shall show that, in expectation, this rank is at most $(b \cdot m/n)^{|\mathtt{BL}(i) \cap [t']|}$, which is at most 1 when $b \cdot m \leq n$ (which is the setting that we shall use when handling Type 3). Hence, we shall be accommodating the treatment of Type 3 at no real cost.

Claim 2.1 (the contribution of a gate of Type 1 (i.e., $i \in T_1$)): *For a uniformly distributed $\overline{I} \in \binom{[n]}{b}^{t'}$ and any $S = S(\overline{I}) \subseteq [b]^{t'}$* (which may depend on $\overline{I}$)*, the expected rank of the matrix that represents the monomials of the bilinear function given by* Eq. (13) *is at most $(b \cdot m/n)^{|\mathtt{BL}(i) \cap [t']|}$.*

Note that $b < n/m$ implies that $(b \cdot m/n)^{d'} \leq 1$, for every $d' \geq 0$, with equality holding only for $d' = 0$.

Proof: Recall that G_i depends only on blocks in $\mathtt{BL}(i)$, whereas F_i depends only on blocks not in $\mathtt{BL}(i)$, and that $t'+1 \in \mathtt{BL}(i)$ but $t'+2 \notin \mathtt{BL}(i)$. Hence, we may replace G_i by the actual function, denoted G'_i, that G_i applies to variables in $\mathtt{BL}(i)$; that is, assuming for simplicity that $\mathtt{BL}(i) = [d'] \cup \{t'+1\}$, where $d' \geq 0$, we define $G'_i(y_1, ..., y_{d'}, y) = G_i(y_1, ..., y_d, 0^n, ..., 0^n, y)$, where it actually does not matter if 0^n is replaced by any other n-bit string.[14] (We could have done the same for F_i, but there is no benefit in doing so.) Then, still using the simplified assumption that $\mathtt{BL}(i) = [d'] \cup \{t'+1\}$, we can re-write Eq. (13) as follows

$$\begin{aligned} &\sum_{(k_1,...,k_{t'}) \in S} F_i(\mathtt{u}(I_1(k_1)), ..., \mathtt{u}(I_{t'}(k_{t'})), x^{(t'+2)}) \cdot G'_i(\mathtt{u}(I_1(k_1)), ..., \mathtt{u}(I_{d'}(k_{d'})), x^{(t'+1)}) \\ &= \sum_{k_1,...,k_{d'} \in [b]} G'_i(\mathtt{u}(I_1(k_1)), ..., \mathtt{u}(I_{d'}(k_{d'})), x^{(t'+1)}) \end{aligned}$$

[14] In general, letting $\mathtt{BL}(i) = \{i_1, ..., i_{d'}, t'+1\}$, where $d' \geq 0$, we define $G'_i(y_1, ..., y_{d'}, y) = G_i(y'_1, ..., y'_{t'}, y)$ such that $y'_{i_j} = y_j$ if $j \in [d']$ and $y'_i = 0^n$ otherwise (i.e., $i \notin \{i_1, ..., i_{d'}\}$).

$$\cdot \sum_{k_{d'+1},...,k_{t'}\in[b]:(k_1,...,k_{t'})\in S} F_i(\mathtt{u}(I_1(k_1)),...,\mathtt{u}(I_{t'}(k_{t'})),x^{(t'+2)})$$

$$= \sum_{k_1,...,k_{d'}\in[b]} G'_i(\mathtt{u}(I_1(k_1)),...,\mathtt{u}(I_{d'}(k_{d'})),x^{(t'+1)})\cdot F_{i,S}^{(k_1,...,k_{d'})}(x^{(t'+2)}) \quad (14)$$

where $F_{i,S}^{(k_1,...,k_{d'})}(z)$ equals the sum

$$\sum_{(k_{d'+1},...,k_{t'})\in S^{(k_1,...,k_{d'})}} F_i(\mathtt{u}(I_1(k_1)),...,\mathtt{u}(I_{t'}(k_{t'})),x^{(t'+2)})$$

where $S^{(k_1,...,k_{d'})} \stackrel{\text{def}}{=} \{(k_{d'+1},...,k_{t'}) \in [b]^{t'-d'} : (k_1,...,k_{t'}) \in S\}$. Looking at Eq. (14), we observe that the term corresponding to $(k_1,...,k_{d'}) \in [b]^{d'}$ (in the sum) vanishes unless for every $j \in [d']$ it holds that the k_j^{th} variable of block $j \in \mathtt{BL}(i) \cap [t']$ feeds G_i. Considering a random choice of $\overline{I}$, the probability that the term corresponding to $(k_1,...,k_{d'}) \in [b]^{d'}$ does not vanish is at most $(m/n)^{d'}$. We stress that this vanishing is due to G_i, and so the fact that $F_{i,S(\overline{I})}^{(k_1,...,k_{d'})}(x^{(t'+2)})$ varies with $\overline{I}$ is immaterial.[15] Hence, the expected number of terms in Eq. (14) that do not vanish is at most $b^{d'}\cdot(m/n)^{d'}$, whereas each term may contribute at most one unit to the rank. It follows that the expected rank of the matrix that represents the bilinear function of Eq. (14) is at most $(b\cdot m/n)^{d'}$, where the expectation is taken uniformly over the choices of $\overline{I} \in \binom{[n]}{b}^{t'}$. ■

Digest (of the proof of Claim 2.1). The key point in the foregoing proof is that both the number of terms in Eq. (14) and the probability that each term does not vanish are exponential in d' (i.e., they are $b^{d'}$ and $(m/n)^{d'}$, respectively), and so the growth of the first is compensated by the decline of the second. Indeed, the fact that $S = S(\overline{I})$ is determined based on $\overline{I}$ is immaterial, since we consider all $b^{d'}$ terms anyhow, and for each term we consider its vanishing due only to the event $G'_i(\mathtt{u}(I_1(k_1)),...,\mathtt{u}(I_{d'}(k_{d'})),x^{(t'+1)}) \equiv 0$.

Conclusion. We believe that the foregoing description would convince most readers of the fact that, for a random sequence $\overline{I} = (I_1,...,I_k)$ of b-subsets, the expected rank of the matrix that represents the bilinear function

$$\sum_{i\in T_1} \sum_{(k_1,...,k_{t'})\in S(\overline{I})} F_i(\mathtt{u}(I_1(k_1)),...,\mathtt{u}(I_{t'}(k_{t'})),x^{(t'+2)})\cdot G_i(\mathtt{u}(I_1(k_1)),...,\mathtt{u}(I_{t'}(k_{t'})),x^{(t'+1)}) \quad (15)$$

is at most $|T_1|\cdot(b\cdot m/n)^{d'} \leq m$, where the inequality uses $b\cdot m \leq n$. Such convinced readers are advised to skip the following paragraph and proceed directly to the treatment of Type 2 gates (provided in Section 2.3.2).

A more formal argument requires some additional notation. For each $i \in T_1$, let $d_i \stackrel{\text{def}}{=} |\mathtt{BL}(i)\cap[t']|$ and $\mathtt{BL}_j(i)$ be the j^{th} element in $\mathtt{BL}(i)\cap[t']$. (In the foregoing simplified argument, we used $d' = d_i$

[15] Recall that $S = S(\overline{I})$ may depends on $\overline{I}$, and that we are considering the term $G'_i(\mathtt{u}(I_1(k_1)),...,\mathtt{u}(I_{d'}(k_{d'})),x^{(t'+1)})\cdot F_{i,S(\overline{I})}^{(k_1,...,k_{d'})}(x^{(t'+2)})$, for a fixed $(k_1,...,k_{d'}) \in [b]^{d'}$ and random $\overline{I}$.

and assumed that $\mathtt{BL}_j(i) = j$.) Then, Eq. (14) is re-written as

$$\sum_{k_{\mathtt{BL}_1(i)},...,k_{\mathtt{BL}_{d_i}(i)} \in [b]} G'_i(\mathtt{u}(I_{\mathtt{BL}_1(i)}(k_{\mathtt{BL}_1(i)})),...,\mathtt{u}(I_{\mathtt{BL}_{d_i}(i)}(k_{\mathtt{BL}_{d_i}(i)})), x^{(t'+1)}) \cdot F_{i,S(\overline{I})}^{(k_{\mathtt{BL}_1(i)},...,k_{\mathtt{BL}_{d_i}(i)})}(x^{(t'+2)}) \quad (16)$$

where G'_i and $F_{i,S}^{(k_1,...,k_{d_i})}$ are defined in an analogous matter.[16] Now, repeating the foregoing argument for each $i \in T_1$, we conclude that, for a random sequence $\overline{I} = (I_1, ..., I_k)$ of b-subsets, the expected rank of the matrix that represents the bilinear function of Eq. (15) is at most $\sum_{i \in T_1} (b \cdot m/n)^{d_i} \leq |T_1| \leq m$, provided that $b \cdot m \leq n$. (This can be seen by using Eq. (16).)[17]

2.3.2 Gates of Type 2: $t'+2 \in \mathtt{BL}(i)$ and $|\mathtt{BL}(i) \cap [t']| \geq d$

Recall that d will be set to equal $\Theta(1/\epsilon)$. We consider the $b^{t'}$ random assignments that were defined in Section 2.3.1; that is, we select a random sequence of b-subsets, denoted $\overline{I} = (I_1, ..., I_{t'})$, and for every $(i_1, ..., i_{t'}) \in I_1 \times \cdots \times I_{t'}$, we consider the assignment $(\mathtt{u}(i_1), ..., \mathtt{u}(i_{t'})) \in \{0,1\}^{t' \cdot n}$, where $\mathtt{u}(i) = 0^{i-1}10^{n-i}$.

Now, analogously to the argument in the proof of Claim 2.1, observe that G_i vanishes unless for every $j \in \mathtt{BL}(i) \cap [t']$ it happens that I_j hits one of the (at most m) variables of block j that feeds G_i. Noting that the probability of this event is at most $(b \cdot m/n)^{|\mathtt{BL}(i) \cap [t']|}$, it follows that G_i vanishes with probability at least $1 - (b \cdot m/n)^{|\mathtt{BL}(i) \cap [t']|}$. Letting T_2 denote the set of all i's of Type 2, it follows that, with probability at least $1 - m \cdot (b \cdot m)^d$, all G_i's of Type 2 vanish, and we can just ignore them, provided that $m \cdot (b \cdot m/n)^d \approx 0$. We shall indeed use a setting that satisfies this (e.g., $m = n^{1-\epsilon}$, $b \leq n^{\epsilon/2}$ and $d \geq 2/\epsilon$). We state this conclusion for sake of future reference.

Claim 2.2 (the contribution of a gate of Type 2 (i.e., $i \in T_2$)): *For a uniformly distributed $\overline{I} \in \binom{[n]}{b}^{t'}$, the probability that G_i does not vanish under some of the assignments specified by $\overline{I}$ is at most $(b \cdot m/n)^{|\mathtt{BL}(i) \cap [t']|} \leq (b \cdot m/n)^d$.*

[16] Specifically, $G'_i(y_1, ..., y_{d_i}, y)$ equals the value of $G_i(y'_1, ..., y'_{t'}, y)$ where $y'_{\mathtt{BL}_j(i)} = y_j$ for every $j \in [d_i]$ and all other y'_k's (which don't effect G_i at all) are set arbitrarily (e.g., to 0^n). Likewise $F_{i,S}^{(k_1,...,k_{d_i})}(z)$ equals the sum

$$\sum_{(k'_1,...,k'_{t'}) \in S : (k'_{\mathtt{BL}_1(i)},...,k'_{\mathtt{BL}_{d_i}(i)}) = (k_1,...,k_{d_i})} F_i(\mathtt{u}(I_1(k'_1)),...,\mathtt{u}(I_{t'}(k'_{t'})), x^{(t'+2)}).$$

[17] Specifically, using

$$\begin{aligned}
&\sum_{i \in T_1} \sum_{(k_1,...,k_{t'}) \in S} F_i(\mathtt{u}(I_1(k_1)),...,\mathtt{u}(I_{t'}(k_{t'})), x^{(t'+2)}) \cdot G_i(\mathtt{u}(I_1(k_1)),...,\mathtt{u}(I_{t'}(k_{t'})), x^{(t'+1)}) \\
&= \sum_{i \in T_1} \sum_{k_{\mathtt{BL}_1(i)},...,k_{\mathtt{BL}_{d_i}(i)} \in [b]} G'_i(\mathtt{u}(I_{\mathtt{BL}_1(i)}(k_{\mathtt{BL}_1(i)})),...,\mathtt{u}(I_{\mathtt{BL}_{d_i}(i)}(k_{\mathtt{BL}_{d_i}(i)})), x^{(t'+1)}) \\
&\qquad \cdot F_{i,S(\overline{I})}^{(k_{\mathtt{BL}_1(i)},...,k_{\mathtt{BL}_{d_i}(i)})}(x^{(t'+2)}),
\end{aligned}$$

observe that for each $i \in T_1$ and each $(k_{\mathtt{BL}_1(i)}, ..., k_{\mathtt{BL}_{d_i}(i)}) \in [b]^{d_i}$, the corresponding term (which contains a multiple of $G'_i(\mathtt{u}(I_{\mathtt{BL}_1(i)}(k_{\mathtt{BL}_1(i)})),...,\mathtt{u}(I_{\mathtt{BL}_{d_i}(i)}(k_{\mathtt{BL}_{d_i}(i)})), x^{(t'+1)})$) does not vanish with probability at most $(m/n)^{d_i}$.

2.3.3 Gates of Type 3: $t'+2 \in \mathtt{BL}(i)$ and $|\mathtt{BL}(i) \cap [t']| < d$

We finally get to the case for which we were preparing all along. Note that we may have m gates of Type 3, and under each random restriction of the foregoing form, each such gate may compute an arbitrary bilinear function over $\Omega(m)$ variables of blocks $t'+1$ and $t'+2$, since it may not be fed by any variables from other blocks (which may lead to its vanishing). Hence, for $m = \Omega(n^{2/3})$, the n-by-n matrix that represents the sum of the m corresponding bilinear functions for each random restriction may be arbitrary. The same holds for the sum of the sums that correspond to several random restrictions, unless there are cancellations between these random restrictions. Indeed, the entire point of choosing $b^{t'}$ random restrictions was to form such cancellations.

Whereas in the previous cases (of handling Types 1 and 2) the action focused on the G_i's, in the current case the action is focused on the F_i's. Specifically, for any possible choice of the random sequence of b-subsets, $\overline{I} = (I_1, ..., I_{t'})$, we consider, for each $(k_1, ..., k_{t'}) \in [b]^{t'}$, the vector $v_{k_1,...,k_{t'}}$ representing the value of each F_i under the assignment $(\mathtt{u}(I_1(k_1)), ..., \mathtt{u}(I_{t'}(k_{t'})))$; that is, the i^{th} entry of $v_{k_1,...,k_{t'}}$ equals $F_i(\mathtt{u}(I_1(k_1)), ..., \mathtt{u}(I_{t'}(k_{t'})))$. Actually, this description suffices only the special case in which for each G_i of Type 3 it holds that $\mathtt{BL}(i) = \{t'+1, t'+2\}$. Assuming this special case as well as $b^{t'} > m$, we may pick a non-empty subset S of these vectors that sums-up to the all-zero vector. Then, for such a set S and for every i (of this type) it holds that

$$\sum_{(k_1,...,k_{t'}) \in S} F_i(\mathtt{u}(I_1(k_1)), ..., \mathtt{u}(I_{t'}(k_{t'}))) = 0, \tag{17}$$

where we rely on the fact that F_i is neither fed by variables of block $t'+1$ (since $t'+1 \in \mathtt{BL}(i)$ be definition) nor fed by variables of block $t'+2$ (since $t'+2 \in \mathtt{BL}(i)$ by the hypothesis of Type 3). This means that these i's do not contribute to the corresponding bilinear function (i.e., Eq. (18)), since the corresponding G_i's are oblivious of the assignment to variables in blocks $[t']$ (by our assumption that $\mathtt{BL}(i) = \{t'+1, t'+2\}$). That is, letting T'_3 denote the set of all gates of this type (i.e., $T'_3 = \{i : \mathtt{BL}(i) = \{t'+1, t'+2\}\}$) and using S as in Eq. (17), the following expression is identically zero.

$$\sum_{i \in T'_3} \sum_{(k_1,...,k_{t'}) \in S} F_i(\mathtt{u}(I_1(k_1)), ..., \mathtt{u}(I_{t'}(k_{t'}))) \cdot G_i(\mathtt{u}(I_1(k_1)), ..., \mathtt{u}(I_{t'}(k_{t'})), x^{(t'+1)}, x^{(t'+2)}) \tag{18}$$

where G_i is actually oblivious of the assignment to the first t' blocks.

Turning to the general case, let T_3 denote the set of all i's of Type 3; that is, $T_3 = \{i : \mathtt{BL}(i) \supseteq \{t'+1, t'+2\} \ \& \ |\mathtt{BL}(i)| < d+2\}$. We shall show that in this case we may pick a non-empty subset S such that a sum analogous to Eq. (18), with T'_3 replaced by T_3, is identically zero. Here we shall use $b^{t'-d} > m$ (rather than $b^{t'} > m$), which is where we use an upper-bound on d.

Claim 2.3 (the contribution of gates of Type 3): *Suppose that $b^{t'-d} \geq m$. Then, for every $\overline{I} \in \binom{[n]}{b}^{t'}$ there exists a set $S = S(\overline{I}) \subseteq [b]^{t'}$ such that for every $i \in T_3$ it holds that*

$$\sum_{(k_1,...,k_{t'}) \in S} F_i(\mathtt{u}(I_1(k_1)), ..., \mathtt{u}(I_{t'}(k_{t'}))) \cdot G_i(\mathtt{u}(I_1(k_1)), ..., \mathtt{u}(I_{t'}(k_{t'})), x^{(t'+1)}, x^{(t'+2)}) = 0. \tag{19}$$

Proof: The problem we face here is that G_i may depend on variables from few blocks in $[t']$; that is, we are only guaranteed that $|\mathtt{BL}(i) \cap [t']| < d$ (whereas for $i \in T'_3$ we used $\mathrm{d}\mathtt{BL}(i) \cap [t'] = \emptyset$).

Hence, establishing Eq. (19) must take into account the fact that G_i may vary among the different terms in the sum (and so establishing Eq. (17) does not suffice).

Intuitively, using the upper bound on $|\mathtt{BL}(i) \cap [t']|$, we can afford to group the terms in Eq. (19) according to the value assigned to the variables in $\mathtt{BL}(i) \cap [t']$, and make sure that each of these $b^{|\mathtt{BL}(i) \cap [t']|}$ sums vanishes. Specifically, letting $d_i \stackrel{\text{def}}{=} |\mathtt{BL}(i) \cap [t']|$ and $\mathtt{BL}_j(i)$ denote the j^{th} element of $\mathtt{BL}(j) \cap [t']$, for every $(k'_1, ..., k'_{d_i}) \in [b]^{d_i}$, we consider the partial sum of Eq. (17) taken over $(k_1, ..., k_{t'}) \in S$ such that $k_{\mathtt{BL}_j(i)} = k'_j$ for every $j \in [d_i]$, and pick S such that all these b^{d_i} partial sums vanish (for all $i \in T_3$). Hence, rather than picking S such that $|T_3|$ equations of the form of Eq. (17) hold, we pick S such that $\sum_{i \in T_3} b^{d_i}$ similar equations hold. The rather straightforward, but tedious, implementation of this idea follows.

Notation (an auxiliary matrix). For any fixed choice of the random t'-long sequence of b-subsets, $\overline{I} = (I_1, ..., I_{t'}) \in \binom{[n]}{b}^{t'}$, we consider an auxiliary $|T_3| \cdot \sum_{d' \in [d-1]} b^{d'}$-by-$b^{t'}$ Boolean matrix in which the rows correspond to pairs $(i, (k'_1, ..., k'_{d'})) \in T_3 \times \bigcup_{d' \in [d-1]} [b]^{d'}$, the columns correspond to choices of $\overline{k} = (k_1, ..., k_{t'}) \in [b]^{t'}$, and the value of the entry $((i, (k'_1, ..., k'_{d'})), \overline{k})$ is determined according to $F_i(\mathtt{u}(I_1(k_1)), ..., \mathtt{u}(I_{t'}(k_{t'})))$, provided that $(k'_1, ..., k'_{d_i}) = \overline{k}_{\mathtt{BL}(i) \cap [t']}$ (i.e., $k'_j = k_{\mathtt{BL}_j(i)}$ for every $j \in [d_i]$). Specifically:

- Recall that, for each $i \in T_3$, we let $d_i \stackrel{\text{def}}{=} |\mathtt{BL}(i) \cap [t']| \leq d - 1$ and $\mathtt{BL}_j(i)$ be the j^{th} element in $\mathtt{BL}(i) \cap [t']$.
- We allocate $\sum_{d' \in [d-1]} b^{d'}$ rows for each $i \in T_3$, but actually use only b^{d_i} rows that correspond to i (while setting each of the other rows to $0^{b^{t'}}$).

  Each of the foregoing b^{d_i} rows will correspond to a choice of $(k'_1, ..., k'_{d_i}) \in [b]^{d_i}$.
- Each column corresponds to a choice of $(k_1, ..., k_{t'}) \in [b]^{t'}$, which represents a choice of a sequence in $I_1 \times \cdots \times I_{t'}$; that is, the choice of $(k_1, ..., k_{t'})$ determines the t'-long sequence $(I_1(k_1), ..., I_{t'}(k_{t'})) \in [n]^{t'}$, which in turn determines the assignment $(\mathtt{u}(I_1(k_1)), ..., \mathtt{u}(I_{t'}(k_{t'})))$ to the first t' blocks.

  (Indeed, while the row-index $(i, (k'_1, ..., k'_{d_i}))$ represents a choice made for the blocks appearing in $\mathtt{BL}(i) \cap [t']$, the column-index $\overline{k} = (k_1, ..., k_{t'})$ represent a choice made for all blocks in $[t']$. Our focus will be on entries $((i, (k'_1, ..., k'_{d_i})), \overline{k})$'s such that $k'_j = k_{\mathtt{BL}_j(i)}$ for all $j \in [d_i]$.)
- The value of entry $((i, (k'_1, ..., k'_{d'})), \overline{k})$ in the matrix equals 1 if and only if
  - $F_i(\mathtt{u}(I_1(k_1)), ..., \mathtt{u}(I_{t'}(k_{t'}))) = 1$, and
  - $d' = d_i$ and $(k'_1, ..., k'_{d_i}) = \overline{k}_{\mathtt{BL}(i) \cap [t']}$; that is, $k'_j = k_{\mathtt{BL}_j(i)}$ for every $j \in [d_i]$.

  Note that this definition relies on the fact that F_i is not fed by variables of block $t' + 2$, which follows from the definition of Type 3 (by which $t' + 2 \in \mathtt{BL}(i)$).

  We stress that the value of entry $((i, (k'_1, ..., k'_{d'})), \overline{k})$ is 0 in case $(k'_1, ..., k'_{d'}) \neq \overline{k}_{\mathtt{BL}(i) \cap [t']}$ (i.e., either $d' \neq d_i$ or $k'_j \neq k_{\mathtt{BL}_j(i)}$ for some $j \in [d_i]$). This means that when considering the row $(i, (k'_1, ..., k'_{d_i}))$ only columns $\overline{k}$ that are "fit" $(i, (k'_1, ..., k'_{d_i}))$ matter.

Suppose, for simplicity, that the foregoing auxiliary matrix contains an all-zero column that is indexed $\overline{k}$. This means that, for every $i \in T_3$, it holds that $F_i(\mathtt{u}(I_1(k_1)), ..., \mathtt{u}(I_{t'}(k_{t'}))) = 0$, because

the $((i,(k_{\mathtt{BL}_1(i)},...,k_{\mathtt{BL}_{d_i}(i)})),\overline{k})$-entry of the matrix is 0 (although this row-index "fits" the column index $\overline{k}$). Using the notation

$$C_i^{(\overline{k})}(y,z) \stackrel{\text{def}}{=} F_i(\mathtt{u}(I_1(k_1)),...,\mathtt{u}(I_{t'}(k_{t'}))) \cdot G_i(\mathtt{u}(I_1(k_1)),...,\mathtt{u}(I_{t'}(k_{t'})),y,z) \tag{20}$$

it follows that in this case $C_i^{(\overline{k})}(y,z)$ is identically zero (for each $i \in T_3$). In this case we establish Eq. (19) by using $S = \{(k_1,...,k_{t'})\}$; that is, S consist of the index of the all-zero column.

In general, the matrix may contain no all-zero columns; still, using $t' \geq d + \log_b m$ (equiv., $b^{t'-d} \geq m$), there is a non-trivial linear combination of the columns that yields an all-zero vector, because there are $b^{t'}$ columns and at most $\sum_{d' \in [d-1]} b^{d'} \cdot m < b^{t'}$ rows. Denoting the set of columns participating in this linear combination by S, for every $i \in T_3$ and $\overline{k'} = (k'_1,...,k'_{d_i}) \in [b]^{d_i}$, it holds that

$$\sum_{\overline{k} \in S : \overline{k}_{\mathtt{BL}(i)} = \overline{k'}} C_i^{(\overline{k})}(y,z) \tag{21}$$

$$= \; G'_i(\mathtt{u}(I_1(k'_1)),...,\mathtt{u}(I_{d_i}(k'_{d_i})),y,z) \cdot \sum_{\overline{k} \in S : \overline{k}_{\mathtt{BL}(i)} = \overline{k'}} F_i(\mathtt{u}(I_1(k_1)),...,\mathtt{u}(I_{t'}(k_{t'}))) \tag{22}$$

where $G'_i(\mathtt{u}(I_1(k'_1)),...,\mathtt{u}(I_{d_i}(k'_{d_i})),y,z)$ equals $G_i(\mathtt{u}(I_1(k_1)),...,\mathtt{u}(I_{t'}(k_{t'})),y,z)$ for every $(k_1,...,k_{t'})$ that satisfies $k_{\mathtt{BL}_j(i)} = k'_j$ for every $j \in [d_i]$. *The key point is that G_i only depends on variables of the blocks in $\mathtt{BL}(i)$, and so G_i is invariant under all assignments that fix the variablres in these blocks.*

The punch-line is that Eq. (22) is identically zero, because the $(i,(k'_1,...,k'_{d_i}))^{\text{th}}$ entry in the corresponding sum of columns is zero, whereas the full row has 0-entries in columns that do not "fit" $(i,(k'_1,...,k'_{d_i}))$, and holds the value of $F_i(\mathtt{u}(I_1(k_1)),...,\mathtt{u}(I_{t'}(k_{t'})))$ in each column $\overline{k}$ that does "fit" $(i,(k'_1,...,k'_{d_i}))$ (i.e., satisfies $\overline{k}_{\mathtt{BL}(i)\cap[t']} = (k'_1,...,k'_{d_i})$). Formally, letting $M_{(i,(k'_1,...,k'_{d_i})),\overline{k}}$ denote the $((i,(k'_1,...,k'_{d_i})),\overline{k})^{\text{th}}$ entry in the foregoing matrix, for every $i \in T_3$ and $\overline{k'} = (k'_1,...,k'_{d_i}) \in [b]^{d_i}$, we have

$$\sum_{\overline{k} \in S : \overline{k}_{\mathtt{BL}(i)} = \overline{k'}} F_i(\mathtt{u}(I_1(k_1)),...,\mathtt{u}(I_{t'}(k_{t'}))) \;=\; \sum_{\overline{k} \in S} M_{(i,(k'_1,...,k'_{d_i})),\overline{k}} \tag{23}$$

$$= \; 0. \tag{24}$$

Combining Eq. (21)&(22) with Eq. (23)&(24), we get (for every $i \in T_3$)

$$\begin{aligned}
\sum_{\overline{k} \in S} C_i^{(\overline{k})}(y,z) &= \sum_{(k'_1,...,k'_{d_i}) \in [b]^{d_i}} \; \sum_{\overline{k} \in S : \overline{k}_{\mathtt{BL}(i)} = \overline{k'}} C_i^{(\overline{k})}(y,z) \\
&= \sum_{(k'_1,...,k'_{d_i}) \in [b]^{d_i}} G'_i(\mathtt{u}(I_1(k'_1)),...,\mathtt{u}(I_{d_i}(k'_{d_i})),y,z) \\
&\quad \cdot \sum_{\overline{k} \in S : \overline{k}_{\mathtt{BL}(i)} = \overline{k'}} F_i(\mathtt{u}(I_1(k_1)),...,\mathtt{u}(I_{t'}(k_{t'}))) \\
&= \sum_{(k'_1,...,k'_{d_i}) \in [b]^{d_i}} G'_i(\mathtt{u}(I_1(k'_1)),...,\mathtt{u}(I_{d_i}(k'_{d_i})),y,z) \cdot 0
\end{aligned}$$

which establishes Eq. (19). The claim follows. ■

Conclusion. Claim 2.3 implies that for every $\overline{I} \in \binom{[n]}{b}^{t'}$ there exists a set $S = S(\overline{I}) \subseteq [b]^{t'}$ such that the Type 3 gates have no contribution to the bilinear function $B_S : GF(2)^{n+n} \to \mathrm{GF}(2)$ defined as

$$B_S(y,z) \stackrel{\text{def}}{=} \sum_{\overline{k}=(k_1,...,k_{t'}) \in S} F(\mathtt{u}(I_1(k_1)),...,\mathtt{u}(I_{t'}(k_{t'})),y,z) \tag{25}$$

where $F = \sum_{i \in [m]:\mathtt{BL}(i) \ni t'+1} F_i \cdot G_i$ is the function supposedly computed by the circuit.

Recall that the contribution of Type 2 gates vanishes with very high probability, and that the contribution of Type 1 gates to $B_{S(\overline{I})}(y,z)$ is represented by a matrix of expected rank at most m, where in both cases the probability space refers to the choice of $\overline{I}$. Hence, with probability at least $2/3$ over the choice of $\overline{I}$, the bilinear function $B_{S(\overline{I})}$ is represented by a matrix of rank $O(m)$. The straightforward but tedious proof of the latter conclusion is detailed in Section 2.4.

2.4 Wrapping-up and reaching a contradiction

We have essentially established the following Lemma 2.4, except that our notations ignored (or hide) the dependence of all residual functions (including $S = S(\overline{I})$) on the values of blocks $t'+3,...,t'+t''$. (Recall that these values were fixed arbitrarily at the very beginning of Section 2.3.)

Lemma 2.4 (low AN2-complexity of F implies low rank of the matrix that represents B_S): *For m, b, t' such that $b \leq (n/m)^{1/2}$ and $t' \geq 2\log_{n/m} n + \log_b m$, suppose that $\mathtt{AN}_2(F) \leq m < n/10$. Then, for every $\overline{s} = (s^{(t'+3)},...,s^{(t'+t'')}) \in \mathrm{GF}(2)^{(t''-2)\cdot n}$, with probability at least $2/3$ over a random choice of $\overline{I} = (I_1,...,I_{t'}) \in \binom{[n]}{b}^{t'}$, there exists a non-empty set $S \subseteq [b]^{t'}$ such that the matrix that represents the bilinear function B_S of* Eq. (25) *has rank at most $5m$. Formally, we refer to the bilinear function*

$$B_S^{(\overline{s},\overline{I})}(y,z) \stackrel{\text{def}}{=} \sum_{\overline{k}=(k_1,...,k_{t'}) \in S} F(\mathtt{u}(I_1(k_1)),...,\mathtt{u}(I_{t'}(k_{t'})),y,z,s^{(t'+3)},....,s^{(t'+t'')}), \tag{26}$$

where $\mathtt{u}(i) = 0^{i-1}10^{n-i}$ is the i^{th} unit vector and $I_j(k)$ denotes the k^{th} element of I_j.

Proof: We merely summarize the contents of Section 2.3, while using the more explicit notations. Recall that our starting point is a depth-two circuit of AN-complexity at most m that computes F, which has a form as captured by Eq. (12). Recall that we have fixed $\overline{s} = (s^{(t'+3)},...,s^{(t'+t'')}) \in \mathrm{GF}(2)^{(t''-2)\cdot n}$ upfront, and the subsequent analysis referred to this fixed $\overline{s}$ (and holds for any such $\overline{s}$). We have broken the sum in Eq. (12) into three parts, corresponding to the three types of gates, and analyzed each type separately.

For Type 1, we showed (in Claim 2.1) that, for a random choice of $\overline{I} = (I_1,....,I_{t'}) \in \binom{[n]}{b}^{t'}$, and for every $S \subseteq [b]^{t'}$ (which may depend on $\overline{I}$), the contribution of gates of Type 1 to the bilinear function $B_S^{(\overline{s},\overline{I})}$ is represented by a matrix of expected rank at most m. (This used $b \leq (n/m)$, which holds under the hypothesis.) Next, we showed (in Claim 2.2) that, over the same random choice of $\overline{I}$, and for every $S \subseteq [b]^{t'}$, the contribution of gates of Type 2 to $B_S^{(\overline{s},\overline{I})}$ is represented by a matrix that is non-zero with probability at most $m \cdot (b \cdot m/n)^d$. Using $b \leq (n/m)^{1/2}$ and setting

$d = 2\log_{n/m} n$, we have $m \cdot (b \cdot m/n)^d \leq m \cdot (m/n)^{d/2} = m/n < 1/10$. Hence, with probability at least $0.8 - 0.1 > 2/3$, the contribution of gates of Types 1 and 2 to $B_S^{(\overline{s},\overline{I})}$ is represented by a matrix of rank at most $5m$.

Lastly, considering the Type 3 gates, we showed (in Claim 2.3) that, for any choice of $\overline{I}$, there exists a non-empty set $S = S(\overline{I}) \subseteq [b]^{t'}$ such that the contribution of functions of Type 3 to $B_S^{(\overline{s},\overline{I})}$ is represented by an all-zero matrix. Here we used $t' \geq d + \log_b m$, which holds by the hypothesis (when using $d = 2\log_{n/m} m$).

Recap. Formally, for $\overline{s} = (s^{(t'+3)}, ..., s^{(t'+t'')})$, and $\overline{I} = (I_1, ..., I_{t'}) \in \binom{[n]}{b}^{t'}$, letting $\overline{\mathtt{u} \circ I}(\overline{k}) = (\mathtt{u}(I_1(k_1)), ..., \mathtt{u}(I_{t'}(k_{t'})))$, we re-write Eq. (26) as

$$B_S^{(\overline{s},\overline{I})}(y,z) = \sum_{\overline{k}=(k_1,...,k_{t'}) \in S} F(\overline{\mathtt{u} \circ I}(\overline{k}), y, z, s^{(t'+3)}, ..., s^{(t'+t'')}). \tag{27}$$

Hence, for the corresponding set S (which may depend on $\overline{s}$ and $\overline{I}$), we have

$$\begin{aligned} B_S^{(\overline{s},\overline{I})}(y,z) &= \sum_{\overline{k} \in S} \sum_{i \in [m]} F_i(\overline{\mathtt{u} \circ I}(\overline{k}), y, z, \overline{s}) \cdot G_i(\overline{\mathtt{u} \circ I}(\overline{k}), y, z, \overline{s}) \\ &= \sum_{\overline{k} \in S} \sum_{i \in T_1} F_i(\overline{\mathtt{u} \circ I}(\overline{k}), y, z, \overline{s}) \cdot G_i(\overline{\mathtt{u} \circ I}(\overline{k}), y, z, \overline{s}) \qquad (28) \\ &\quad + \sum_{\overline{k} \in S} \sum_{i \in T_2} F_i(\overline{\mathtt{u} \circ I}(\overline{k}), y, z, \overline{s}) \cdot G_i(\overline{\mathtt{u} \circ I}(\overline{k}), y, z, \overline{s}), \qquad (29) \end{aligned}$$

since the sum that corresponds to T_3 is identically zero due to the choice of S. The lemma follows by recalling that, with probability at least $4/5$, the matrix representing the monomials of the bilinear function captured by Eq. (28) has rank at most $5m$, and that, with probability at least $9/10$, the matrix representing the monomials of the bilinear function captured by Eq. (29) is identically zero. ■

Reaching a contradiction. Lemma 2.4 implies that, with probability at least $2/3$ over a random choice of $\overline{s} = (s^{(t'+3)}, ..., s^{(t'+t'')}) \in \mathrm{GF}(2)^{(t''-2)\cdot n}$ and $\overline{I} = (I_1, ..., I_{t'}) \in \binom{[n]}{b}^{t'}$, there exists a non-empty set $S \subseteq [b]^{t'}$ such that the matrix that represents the bilinear function $B_S^{(\overline{s},\overline{I})}$ has rank at most $5 \cdot \mathtt{AN}_2(F)$, provided that $b \leq (n/\mathtt{AN}_2(F))^{1/2}$ and $t' \geq 2\log_{n/\mathtt{AN}_2(F)} n + \log_b \mathtt{AN}_2(F)$. In contrast, the following Lemma 2.5 implies that with probability at least $1 - b^{t'} \cdot 2^{-n/2}$ over the same random choices, for every non-empty set $S \subseteq [b]^{t'}$, the matrix that represents the bilinear function $B_S^{(\overline{s},\overline{I})}$ has rank $\Omega(n)$. Hence, we reach contradiction unless either $\mathtt{AN}_2(F) = \Omega(n)$ or $2^{b^{t'}} \cdot 2^{-n/2} > 1/3$ (for b and t' as above). As detailed below, this implies $\mathtt{AN}_2(F) = \Omega(n^{1-\epsilon})$ for $\epsilon = O(1/\sqrt{t'})$, and Theorem 1.8 follows. But let us first prove the following lemma.

Lemma 2.5 (typically the matrix that represents B_S has high rank): *Suppose that the generator $G_{\mathtt{sb}} : \mathrm{GF}(2)^{(t''-2)\cdot n} \to \{0,1\}^{n^{t'+2}}$ has bias at most 2^{-n}. Then, for every sequence of b-subsets $\overline{I} = (I_1, ..., I_{t'}) \in \binom{[n]}{b}^{t'}$ and any non-empty set $S \subseteq [b]^{t'}$, with probability $1 - 2^{-n/2}$ over the choice of $\overline{s} = (s^{(t'+3)}, ..., s^{(t'+t'')}) \in \mathrm{GF}(2)^{(t''-2)\cdot n}$, it holds that the matrix that represents the bilinear function $B_S^{(\overline{s},\overline{I})}$ of* Eq. (27) *has rank $\Omega(n)$.*

Proof: For every $S \subseteq [b]^{t'}$, $\overline{I} \in \binom{[n]}{b}^{t'}$ and $\overline{s} = (s^{(t'+3)}, ..., s^{(t'+t'')}) \in \mathrm{GF}(2)^{(t''-2)\cdot n}$, looking at the value of $B_S^{(\overline{s},\overline{I})}$, while letting $\overline{\mathtt{u}\circ I}(\overline{k}) = (\mathtt{u}(I_1(k_1)), ..., \mathtt{u}(I_{t'}(k_{t'})))$, observe that

$$\begin{aligned} B_S^{(\overline{s},\overline{I})}(y,z) &= \sum_{\overline{k}\in S} F(\overline{\mathtt{u}\circ I}(\overline{k}), y, z, \overline{s}) \\ &= \sum_{\overline{k}\in S} \sum_{(i_1,...,i_{t'+2})\in \overline{I}\times[n]^2} G_{\mathtt{sb}}(\overline{s})_{(i_1,...,i_{t'+2})} \cdot \left(\prod_{j\in[t']} \mathtt{u}(I_j(k_j))_{i_j}\right) \cdot y_{i_{t'+1}} \cdot z_{i_{t'+2}} \end{aligned} \quad (30)$$

where $\mathtt{u}(I_j(k_j))_{i_j}$ denotes the i_j^{th} element in $\mathtt{u}(I_j(k_j))_{i_j}$, and the last equality uses Eq. (8). Observing that, for each $\overline{k} \in S$, only the t'-tuple $(i_1, ..., i_{t'})$ that satisfies $i_j = I_j(k_j)$ (for every $j \in [t']$) contribute to Eq. (30), we get

$$B_S^{(\overline{s},\overline{I})}(y,z) = \sum_{\overline{k}\in S} \sum_{(i_{t'+1},i_{t'+2})\in[n]} G_{\mathtt{sb}}(\overline{s})_{(I_1(k_1),...,I_{t'}(k_{t'}),i_{t'+1},i_{t'+2})} \cdot y_{i_{t'+1}} \cdot z_{i_{t'+2}} \quad (31)$$

The difference between Eq. (30) and Eq. (31) is that in the latter form it is evident that the corresponding matrix is a non-zero linear combination of $|S|$ matrices that correspond to disjoint parts of the output of the small-bias generator $G_{\mathtt{sb}}$; that is, the $(i_{t'+1}, i_{t'+2})^{\text{th}}$ element in the matrix that corresponds to $\overline{k} = (k_1, ..., k_{t'}) \in S$ equals the $(I_1(k_1), ..., I_{t'}(k_{t'}), i_{t'+1}, i_{t'+2})^{\text{th}}$ bit in the output of $G_{\mathtt{sb}}$.

Hence, for any fixed S and $\overline{I}$, when $\overline{s}$ is uniformly distributed in $\{0,1\}^{(t''-2)\cdot n}$, the matrix that represents $B_S^{(\overline{s},\overline{I})}$ is an n-by-n matrix whose entries are distributed according to an 2^{-n}-bias sequence, because any sequence that is obtained by taking linearly independent non-zero linear-combinations of elements in an ϵ-bias sequence is itself ϵ-bias. The lemma follows by using the fact that, with probability at least $1-2^{-n/2}$, such a matrix has rank $\Omega(n)$. Specifically, we upper-bound the probability that the matrix has rank at most $n/10$ by considering all linear combinations of up to $n/10$ columns. Each such linear combination results in an n-long 2^{-n}-bias sequence, and the probability that such a sequence equals the all-zero sequence is at most $2^{-n} + 2^{-n}$.[18] Hence, the probability of the bad event is upper-bounded by $\sum_{i\in[n/10]} \binom{n}{i} \cdot 2^{-n+1} < 2^{-n/2}$, and the lemma follows. ■

Conclusion (re-iterated and detailed): Using a union bound on all possible $S \subseteq [b]^{t'}$, Lemma 2.5 implies that, with probability at least $1 - 2^{b^{t'}} \cdot 2^{-n/2}$ over the choices of $\overline{s} = (s^{(t'+3)}, ..., s^{(t'+t'')}) \in \mathrm{GF}(2)^{(t'-2)\cdot n}$ and $\overline{I} = (I_1, ..., I_{t'}) \in \binom{[n]}{b}^{t'}$, for every non-empty $S \subseteq [b]^{t'}$, the matrix that represents $B_S^{(\overline{s},\overline{I})}$ has rank $\Omega(n)$. On the other hand, Lemma 2.4 implies that, under the same probability space, with probability at least $2/3$, there exists a set S such that the matrix that represents $B_S^{(\overline{s},\overline{I})}$ has rank at most $5 \cdot \mathtt{AN}_2(F)$. Hence, we reach contradiction unless either $\mathtt{AN}_2(F) = \Omega(n)$ or $2^{b^{t'}} \cdot 2^{-n/2} > 1/3$. Using $b = (n/\mathtt{AN}_2(F))^{\beta}$, for $\beta \leq 1/2$, and setting $t' = 2\log_{n/\mathtt{AN}_2(F)} n + \log_b \mathtt{AN}_2(F)$,

[18] The max-norm difference between the resulting distribution and the uniform one is upper-bounded by the difference according to the L2-norm, which equals the bias of the sequence (cf., [2, Sec. 1.5]).

we get

$$\begin{aligned} b^{t'} &= b^{2\log_{n/\mathtt{AN}_2(F)} n + \log_b \mathtt{AN}_2(F)} \\ &= n^{2\beta} \cdot \mathtt{AN}_2(F) \end{aligned}$$

which means that $2^{b^{t'}} \cdot 2^{-n/2} > 1/3$ holds if and only if $n^{2\beta} \cdot \mathtt{AN}_2(F) > 0.5 \cdot n - \log_2 3$. Hence, we reach contradiction unless $\mathtt{AN}_2(F) > 0.5 \cdot n^{1-2\beta} - o(1)$. Setting $\beta = \epsilon/2$, we conclude that $\mathtt{AN}_2(F) = \Omega(n^{1-\epsilon})$, with $t' \leq 2\log_{n^\epsilon} n + \log_{(n^\epsilon)^\beta} n = O(1/\epsilon^2)$. Theorem 1.8 follows by using an adequate small-bias generator, as provided by Theorem A.6.

Remark 2.6 (using rigidity rather than pure rank): *We note that the proof of Lemma 2.4 can be adapted to show that the corresponding matrix has rigidity $O(b^d \cdot m^{d+3}/n^d)$ with respect to rank $5m$. On the other hand, using [3, Footnote 13], one can adapt the proof of Lemma 2.5 to show that the corresponding matrix has rigidity $\Omega(n^3/m^2)$ with respect to rank $5m$. We stress that, like the argument regarding rank, the argument regarding rigidity requires $m \cdot b^d < 0.5n - 2$ and $b \geq 2$. Using $b = (n/m)^\beta$, the rigidity argument allows to infer that $m = \Omega(n^{1-\frac{2}{(1-\beta)\cdot d+5}})$ rather than $m = \Omega(n^{1-2\beta})$ as inferred by the foregoing rank argument, when using $d = 2/\epsilon = 1/\beta$. Hence, obtaining $m = \Omega(n^{1-\epsilon})$ via the rigidity argument uses d such that $\frac{2}{(1-\beta)\cdot d+5} = \epsilon$, which yields $d = \frac{2}{\epsilon} - \frac{8}{2-\epsilon}$. This modest gain* (of approximately four units) *in d translates to a similar gain in t'.*

The foregoing comparison refers to the current setting of d and b, which is not optimal anyhow. But it seems that $t' = \Omega(1/\epsilon^2)$ will follow in any case.

3 Proof of Theorem 1.9

We adapt the techniques used in Section 2 in order to prove the same lower bound on a more explicit function. Specifically, for t' to be determined, we consider the $(t'+3)$-linear function $\mathtt{f} : \mathrm{GF}(2)^{(t'+2)\cdot n + (t'+2)n} \to \mathrm{GF}(2)$ defined as

$$\mathtt{f}(x^{(1)}, x^{(2)}, ..., x^{(t'+3)}) = \sum_{i_1,...,i_{t'+2} \in [n]} \left(\prod_{j \in [t'+2]} x^{(j)}_{i_j} \right) \cdot x^{(t'+3)}_{i_1+i_2+\cdots+i_{t'+2}} \tag{32}$$

where the last block of variables has length $(t'+2) \cdot n$ rather than n. (Alternatively, we may partition the last block to $t'+2$ blocks holding n variables each.)[19] We mention that this function generalized the trilinear function of Part 2 of Theorem 1.6.

The analysis of depth-two multilinear circuits, which underlies the proof of Lemma 2.4 (i.e., the "low AN2-complexity implies low rank"), remains almost intact, whereas the "high rank lemma" is totally different. In order to fit the latter lemma, the "low rank lemma" is restricted in the choice of a non-empty set $S \subseteq [b]^{t'}$ (such that the matrix that represents the bilinear function B_S has low rank). Specifically, rather than asserting (for every $\overline{I} = (I_1, ..., I_{t'})$) the existence of an arbitrary non-empty set S in $[b]^{t'}$, we consider only non-empty sets S such that for every v there exists at most

[19] That is, the variable-block $x^{(t'+2)} = (x^{(t'+2)}_1, ..., x^{(t'+2)}_{(t'+2)n})$ is replaced by $t'+2$ variables-blocks $(x^{(t'+3)}, x^{(t'+4)}, ..., x^{(2t'+4)})$ such that the $(j-1)\cdot n + k^{\text{th}}$ bit of $x^{(t'+3)}$ is replaced by the k^{th} bit of $x^{(t'+2+j)}$. Strictly speaking, this yields a multilinear function that does not fit Eq. (1), but this can be corrected by augmenting each variable-block with a dummy variable that will be set to 1.

one $(k_1, ..., k_{t'}) \in S$ such that $\sum_{j\in[t']} I_j(k_j) = v$ (equiv., $|\{\sum_{j\in[t']} I_j(k_j) : (k_1, ..., k_{t'}) \in S\}| = |S|$). That is, defining $\mu_{\overline{I}} : [b]^{t'} \to [t'n]$ such that

$$\mu_{\overline{I}}(k_1, ..., k_{t'}) \stackrel{\text{def}}{=} \sum_{j\in[t']} I_j(k_j), \tag{33}$$

we say that $S \subseteq [b]^{t'}$ is $\overline{I}$-admissible if $|\{\mu_{\overline{I}}(\overline{k}) : \overline{k} \in S\}| = |S|$, which means that the function $\mu_{\overline{I}}$ is injective when restricted to S. With this definition in place, we adapt the "low rank lemma" as follows.

Lemma 3.1 (low AN2-complexity of $\mathtt{f}$ implies low rank of the matrix that represents B_S): *For m, b, t' such that $b \in [\omega(1), (n/m)^{1/2}]$ and $t' \in [2\log_{n/m} n + \log_b m + 1, \log_b(n/2) - 1]$, suppose that $\mathtt{AN}_2(\mathtt{f}) \leq m < n/10$. Then, for every $w \in \mathrm{GF}(2)^{(t'+2)\cdot n}$, with probability at least $2/3$ over a random choice of $\overline{I} = (I_1, ..., I_{t'}) \in \binom{[n]}{b}^{t'}$, there exists a non-empty $\overline{I}$-*admissible *set $S \subseteq [b]^{t'}$ such that the matrix that represents the following bilinear function $B_S^{(w,\overline{I})}$ has rank at most $5m$.*

$$B_S^{(w,\overline{I})}(y, z) \stackrel{\text{def}}{=} \sum_{\overline{k}=(k_1,...,k_{t'})\in S} \mathtt{f}(\mathtt{u}(I_1(k_1)), ..., \mathtt{u}(I_{t'}(k_{t'})), y, z, w), \tag{34}$$

where $\mathtt{u}(i) = 0^{i-1}10^{n-i}$ is the i^{th} unit vector and $I_j(k)$ denotes the k^{th} element of I_j.

Recall that we were using $b^{t'} < n/2$ (equiv., $t' < \log_b(n/2) - 1$) anyhow (when contrasting the "low rank lemma" with the "high rank lemma"), and so we lose nothing by the restriction $t' \leq \log_b(n/2) - 1$. Ditto regarding the condition $b = \omega(1)$, which is actually not essential (i.e., $b \geq C$ for a sufficiently large constant C will do).

Proof: We note that the proof of Lemma 2.4 actually establishes its claim with probability at least 0.7 (rather than 2/3) over the choice of $\overline{I}$, but it does not necessarily select an $\overline{I}$-admissible set S. Recall that the existence of a suitable set $S \subseteq [b]^{t'}$ is merely based on the fact that $|[b]^{t'}| \geq b^d \cdot m$, and any subset U of $[b]^{t'}$ that has size at least $b^d \cdot m$ will do (i.e., allow us to argue that there exists an adequate set $S \subseteq U$ that satisfies the claim of the lemma).

Now, using the hypothesis $b^{t'} \leq n/2b$, we show that, *with probability $1 - o(1)$ over the choice of $\overline{I}$, the set $[b]^{t'}$ contains an $\overline{I}$-admissible set U of size at least $b^{t'}/2$*, and so we can afford to restrict S to be a subset of U (since $b^{t'-1} \geq b^d \cdot m$). To prove the foremer fact, observe that, for every two distinct sequences $\overline{k} \neq \overline{k}'$ in $[b]^{t'}$, it holds that $\Pr_{\overline{I}}[\mu_{\overline{I}}(\overline{k}) = \mu_{\overline{I}}(\overline{k}')] \leq 1/(n-1)$, because $\Pr_{I\in\binom{[n]}{b}}[I(k) = I(k')] = 1/(n-1)$ for distinct $k, k' \in [b]$. This implies that for every $\overline{k} \in [b]^{t'}$, we have

$$\Pr_{\overline{I}}[|\mu_{\overline{I}}^{-1}(\mu_{\overline{I}}(\overline{k}))| > 1] \;\leq\; \sum_{\overline{k}'\in[b]^{t'}\setminus\{\overline{k}\}} \Pr_{\overline{I}}[\mu_{\overline{I}}(\overline{k}') = \mu_{\overline{I}}(\overline{k})] \;\leq\; \frac{b^{t'} - 1}{n - 1} \tag{35}$$

which is smaller than $1/2b$ since $t' \leq \log_b(n/2) - 1$. The claim follows, because the probability of having more than half of the sequences $\overline{k} \in [b]^{t'}$ violate the event at the l.h.s. of Eq. (35) is at most $1/b$. ■

The new "high rank lemma". The key observation is that, for every fixed $(i_1,...,i_{t'}) \in [n]^{t'}$, the residual function $\mathtt{f}'(y,z,w) \stackrel{\text{def}}{=} \mathtt{f}(\mathtt{u}(i_1),...,\mathtt{u}(i_{t'}),y,z,w)$ is closely related to the trilinear function of Part 2 of Theorem 1.6. In particular, for a uniformly distributed $w \in \mathrm{GF}(2)^{(t'+2)n}$, the matrix that that represents the residual bilinear function $\mathtt{f}'(\cdot,\cdot,w)$ is a random Toeplitz (or rather Hankel) matrix. Furthermore, the same holds for $B_S^{(w,\overline{I})}$ provided that the set S is $\overline{I}$-admissible (and non-empty). These observation allow us to prove the following lemma.

Lemma 3.2 (typically the matrix that represents B_S has high rank): *For every sequence of b-subsets $\overline{I} = (I_1,...,I_{t'}) \in \binom{[n]}{b}^{t'}$ and any non-empty and $\overline{I}$-*admissible *set $S \subseteq [b]^{t'}$, with probability $1-2^{-n/2}$ over the choice of $w \in \mathrm{GF}(2)^{(t'+2)\cdot n}$, it holds that the matrix that represents the bilinear function $B_S^{(w,\overline{I})}$ of* Eq. (34) *has rank $\Omega(n)$.*

Proof: For every $S \subseteq [b]^{t'}$, $\overline{I} \in \binom{[n]}{b}^{t'}$ and $w \in \mathrm{GF}(2)^{(t'+2)\cdot n}$, looking at the value of $B_S^{(w,\overline{I})}$, while letting $\overline{\mathtt{u}\circ I}(\overline{k}) = (\mathtt{u}(I_1(k_1)),...,\mathtt{u}(I_{t'}(k_{t'})))$, we have

$$\begin{aligned}
B_S^{(w,\overline{I})}(y,z) &= \sum_{\overline{k}\in S} \mathtt{f}(\overline{\mathtt{u}\circ I}(\overline{k}),y,z,w) \\
&= \sum_{\overline{k}\in S} \sum_{i_1,...,i_{t'+2}\in[n]} \cdot \left(\prod_{j\in[t']} \mathtt{u}(I_j(k_j))_{i_j}\right) \cdot y_{i_{t'+1}} \cdot z_{i_{t'+2}} \cdot w_{i_1+i_2+\cdots+i_{t'}+i_{t'+1}+i_{t'+2}} \\
&= \sum_{\overline{k}\in S} \sum_{i_{t'+1},i_{t'+2}\in[n]} y_{i_{t'+1}} \cdot z_{i_{t'+2}} \cdot w_{I_1(k_1)+I_2(k_2)+\cdots+I_{t'}(k_{t'})+i_{t'+1}+i_{t'+2}}
\end{aligned}$$

where the second equality is due to Eq. (32), and the third equality is due to the fact that assigning $\overline{\mathtt{u}\circ I}(\overline{k})$ to the first t' variable-blocks eliminates all terms in the inner sum that are not indexed by $(I_1(k_1),I_2(k_2),...,I_{t'}(k_{t'}),\cdot,\cdot)$, because $\prod_{j\in[t']} \mathtt{u}(I_j(k_j))_{i_j} = 0$ if and only if there exists $j \in [t']$ such that $i_j \neq k_j$. Using Eq. (33) (i.e., $\sum_{j\in[t']} I_j(k_j)) = \mu_{\overline{I}}(k_1,k_2,...,k_{t'})$), we obtain

$$B_S^{(w,\overline{I})}(y,z) = \sum_{i_{t'+1},i_{t'+2}\in[n]} y_{i_{t'+1}} \cdot z_{i_{t'+2}} \cdot \sum_{\overline{k}\in S} w_{\mu_{\overline{I}}(k_1,...,k_{t'})+i_{t'+1}+i_{t'+2}} \tag{36}$$

Using the hypothesis that S is $\overline{I}$-admissible, we observe that the w-variables in the inner sum are distinct; that is, for every $i_{t'+1},i_{t'+2} \in [n]$, different $\overline{k}$'s yield different values of $\mu_{\overline{I}}(\overline{k}) + i_{t'+1} + i_{t'+2}$, which implies that there are no cancellation among terms of the inner sum. Indeed, the key observation is that, for every $i_{t'+1},i_{t'+2} \in [n]$, the sum $\sum_{\overline{k}\in S} w_{\mu_{\overline{I}}(\overline{k})+i_{t'+1}+i_{t'+2}}$ does not vanish when $S \neq \emptyset$ is $\overline{I}$-admissible. Using $S' = \{\mu_{\overline{I}}(\overline{k}) : \overline{k} \in S\}$ (along with the foregoing observation), Eq. (36) implies that that there exists a non-empty set $S' \subseteq [(t'+2)\cdot n]$ such that

$$B_S^{(w,\overline{I})}(y,z) = \sum_{i_{t'+1},i_{t'+2}\in[n]} y_{i_{t'+1}} \cdot z_{i_{t'+2}} \cdot \sum_{s\in S'} w_{s+i_{t'+1}+i_{t'+2}} \tag{37}$$

Note that the bilinear function of Eq. (37) is represented by a Hankel matrix, since it has the form $\sum_{i,j\in[n]} w'_{i+j}\cdot y_i \cdot z_j$ such that $w'_{i,j}$ is invariant when $i+j$ is fixed. Specifically, we consider the n-by-n matrix $(w'_{i,j})_{i,j\in[n]}$ such that $w'_{i,j} = \sum_{s\in S'} w_{s+i+j}$. Furthermore, when selecting $w \in \mathrm{GF}(2)^{(t'+2)n}$

uniformly at random, the bilinear function of Eq. (37) is represented by a random n-by-n Hankel matrix; that is, the sequence $(\sum_{s\in S'} w_{s+2}, ..., \sum_{s\in S'} w_{s+2n})$ is uniformly distributed in $\mathrm{GF}(2)^{2n-1}$. This is the case because the mapping $w \mapsto (\sum_{s\in S'} w_{s+2}, ..., \sum_{s\in S'} w_{s+2n})$ is linear and onto.[20]

Finally, note that a random n-by-n Hankel matrix has rank at least $n/10$ with probability at least $1 - 2^{-n/2}$. This is the case since each (non-trivial) linear combination of its columns is uniformly distributed in $\mathrm{GF}(2)^n$, and so the probability that there exists a linear combination of at most $n/10$ columns that equals the all-zero vector is upper-bounded by $\sum_{i\in[n/10]} \binom{n}{i} \cdot 2^{-n} < 2^{-n/2}$. The lemma follows. ■

Conclusion: As in Section 2.4, using a union bound on all possible $S \subseteq [b]^{t'}$, Lemma 3.2 implies that, with probability at least $1 - 2^{b^{t'}} \cdot 2^{-n/2} > 1/2$ over the choices of $w \in \mathrm{GF}(2)^{(t'+2)\cdot n}$ and $\overline{I} = (I_1, ..., I_{t'}) \in \binom{[n]}{b}^{t'}$, for every non-empty and $\overline{I}$-admissible $S \subseteq [b]^{t'}$, the matrix that represents $B_S^{(w,\overline{I})}$ has rank $\Omega(n)$. On the other hand, Lemma 3.1 implies that, under the same probability space, with probability at least $2/3$, there exists a non-empty and $\overline{I}$-admissible set S such that the matrix that represents $B_S^{(w,\overline{I})}$ has rank at most $5 \cdot \mathtt{AN}_2(\mathtt{f})$. Hence, we reach contradiction unless $\mathtt{AN}_2(\mathtt{f}) \geq n^{1-\epsilon}$, but this is all conditioned on $2^{b^{t'}} \cdot 2^{-n/2} < 1/2$. Using $m = n^{1-\epsilon}$ and $b = (n/m)^{0.4\epsilon}$, and setting $t' = 2\log_{n/m} n + \log_b m + 1$, the foregoing condition holds (since $b^{t'} = n^{0.8\epsilon} \cdot m \cdot b = o(n)$). We conclude that $\mathtt{AN}_2(\mathtt{f}) \geq n^{1-\epsilon}$, where $t' = 2\log_{n^\epsilon} n + \log_{(n^\epsilon)^{0.4\epsilon}} n + 1 = 2 \cdot \epsilon^{-1} + (0.4 \cdot \epsilon^2)^{-1} + 1$. Noting that $(2\epsilon + (0.4)^{-1}) \cdot \epsilon^{-2} + 1 < 4 \cdot \epsilon^{-2} - 3$ holds for $\epsilon \leq 1/4$, we get $t' + 3 \leq 4/\epsilon^2$, and Theorem 1.9 follows.

Remark 3.3 (computing $\mathtt{f}$ in quadratic time): *For every $t, n \in \mathbb{N}$ and $s \in [t, tn]$, consider the function $F_s^{(t)} : \{0,1\}^{tn} \to \{0,1\}$ defined by*

$$F_s^{(t)}(x^{(1)}, x^{(2)}, ..., x^{(t)}) = \sum_{i_1,...,i_t \in [n]: \sum_{j\in[t]} i_j = s} \prod_{j\in[t]} x_{i_j}^{(j)} \tag{38}$$

and note that $F_s^{(t)}(x^{(1)}, x^{(2)}, ..., x^{(t)})$ equals $\sum_{s'\in[s-n,s-1]} F_{s'}^{(t-1)}(x^{(1)}, x^{(2)}, ..., x^{(t-1)}) \cdot x_{s-s'}^{(t)}$. Hence, the sequence $(F_s^{(t)})_{s\in[t,tn]}$ can be computed in time $t \cdot tn \cdot n$ (by Dynamic programming). *Observing that $\mathtt{f}(x^{(1)}, x^{(2)}, ..., x^{(t'+3)})$ equals $\sum_{s\in[t'+2,(t'+2)n]} F_s^{(t'+2)}(x^{(1)}, x^{(2)}, ..., x^{(t'+2)}) \cdot x_s^{(t'+3)}$, it follows that* Eq. (32) *can be computed in quadratic-time.*

In contrast, it is not clear whether the functions used in the proof of Theorem 1.8 can be computed in fixed polynomial-time, rather than in $O(n^{\mathrm{poly}(1/\epsilon)})$-time.

4 Extension to vanishing ϵ

Theorem 1.8 (and likewise Theorem 1.9) can be extended to $\epsilon = \epsilon(n)$ that vanishes with n, provided that $\epsilon(n) \geq \sqrt{2/\log_2 n}$ (resp., $\epsilon(n) = \omega(\sqrt{1/\log_2 n})$), since the argument conditions ϵ only by presupposing that $b = n^{\epsilon\cdot\beta} = n^{\epsilon^2/2}$ is at least 2 (resp., $\omega(1)$). Hence, we actually have

[20] This can be seen by considering the corresponding linear system $\sum_{s\in S'} x_{s+i} = b_i$ for $i = 2, ..., 2n$, and observing that the corresponding $(2n-1)$-by-$|\{s+i : s\in S' \,\&\, i\in[2,2n]\}|$ matrix has full rank (due to columns $s+2, ..., s+2n$, where s is the largest element in S').

Theorem 4.1 (Theorem 1.8, rephrased): *For every* $\epsilon : \mathbb{N} \to (0,1)$ *such that* $\epsilon(n) \geq \sqrt{2/\log_2 n}$*, letting* $t(n) = \mathrm{poly}(1/\epsilon(n))$*, there exists a quasi-polynomial-time computable* $t(n)$*-linear function* $f : \{0,1\}^{t(n)\cdot n} \to \{0,1\}$ *such that* $\mathtt{AN}_2(f) = \Omega(n^{1-\epsilon(n)})$.

An analogous result holds for Theorem 1.9, where we may use $t(n) = O(1/\epsilon(n)^2)$ and have quadratic-time. Note that the foregoing does not allow setting $\epsilon(n) = \Omega(1/\log n)$ and deriving an $\Omega(n)$ lower bound, and even such a lower bound would have missed the target of being truly linear in the length of the input (i.e., $\mathrm{poly}(1/\epsilon(n)) \cdot n$). This raises several challenges:

1. Prove that there exists a quasi-polynomial-time computable $\mathrm{poly}(\log n)$-linear function $f : \{0,1\}^{\widetilde{O}(n)} \to \{0,1\}$ such that $\mathtt{AN}_2(f) = \Omega(n)$.
2. Improving over Item 1, prove that such a function can be computed in polynomial-time.
3. Improving over Item 1, for $t(n) = \mathrm{poly}(\log n)$, prove that there exists a quasi-polynomial-time computable $t(n)$-linear function $f : \{0,1\}^{t(n)\cdot n} \to \{0,1\}$ such that $\mathtt{AN}_2(f) = \Omega(t(n) \cdot n)$.

   As in Item 2, improve the running time to polynomial.

Of course, a more important challenge is to address the first part of Problem 1.7; that is, presenting an explicit $O(1)$-linear function $f : \{0,1\}^{O(n)} \to \{0,1\}$ satisfying $\mathtt{AN}(f) = \omega(n^{2/3})$, let alone $\mathtt{AN}(f) = \Omega(n^{0.99})$. Recall that the general AN-complexity of multi-linear functions may be lower than its depth-two complexity; in fact, there exist bilinear functions f such that $\mathtt{AN}(f) = o(\mathtt{AN}_2(f))$; specifically, $\mathtt{AN}(f) = O(n^{1/2})$ and $\mathtt{AN}_2(f) = \Omega(n^{2/3})$ [4, Thm 2.3].

Acknowledgements

I am deeply indebted to Avishay Tal for finding a flaw in my original argument and showing me how to fix it. In my opinion, his contribution to the current work fully justifies his co-authoring it, but he refused to do so.

I am also grateful to Benny Applebaum for information and advice regarding the construction of small-bias generators in the current setting.

I am indebted to Avi Wigderson for a discussion that led to the phrasing and proving of Theorem 1.9.

Work done while visiting the Computer Science Department of Columbia University. This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No. 819702).

References

[1] O. Goldreich. *Computational Complexity: A Conceptual Perspective*. Cambridge University Press, 2008.

[2] O. Goldreich. Three XOR-Lemmas: An Exposition. In *Studies in Complexity and Cryptography*, Lecture Notes in Computer Science (Vol. 6650), Springer, 2011. Preliminary version in *ECCC*, TR95-056, 1995.

[3] O. Goldreich and A. Tal. Matrix rigidity of random Toeplitz matrices. *Computational Complexity*, Vol. 27 (2), pages 305–350, 2018. Preliminary versions in *48th STOC* (2016) and *ECCC* TR15-079 (2015).

[4] O. Goldreich and A. Wigderson. On the Size of Depth-Three Boolean Circuits for Computing Multilinear Functions. In *Computational Complexity and Property Testing*, Lecture Notes in Computer Science (Vol. 12050), Springer, 2020. Preliminary version in *ECCC*, TR13-043, 2013.

[5] S. Jukna. *Boolean Function Complexity: Advances and Frontiers*. Algorithms and Combinatorics, Vol. 27, Springer, 2012.

[6] E. Mossel, A. Shpilka, and L. Trevisan. On epsilon-biased generators in NC0. *Random Structures and Algorithms*, Vol. 29 (1), pages 56–81, 2006. Preliminary version in *44th FOCS*, 2003.

[7] J. Naor and M. Naor. Small-bias Probability Spaces: Efficient Constructions and Applications. *SIAM Journal on Computing*, Vol 22, 1993, pages 838–856, 1993. Preliminary version in *22nd STOC*, 1990.

[8] N. Nisan and A. Wigderson. Lower Bound on Arithmetic Circuits via Partial Derivatives. *Computational Complexity*, Vol. 6, pages 217–234, 1996.

[9] L.G. Valiant. Graph-theoretic arguments in low-level complexity. Mathematical Foundations of Computer Science, Lecture Notes in Computer Science (Vol. 53), pages 162–176, Springer, 1977.

[10] L.G. Valiant. Exponential lower bounds for restricted monotone circuits. In *15th ACM Symposium on the Theory of Computing*, pages 110–117, 1983.

[11] E. Viola. The Sum of D Small-Bias Generators Fools Polynomials of Degree D. *Computational Complexity*, Vol. 18 (2), pages 209–217, 2009. Preliminary version in *ECCC*, TR07-132, 2007.

Appendix: On small-bias generators of large stretch

In this appendix we present explicit constructions of small-bias generators of arbitrary large (polynomial) stretch that can be computed by $O(1)$-linear functions. Our start point is the generator of of Mossel, Shpilka, and Trevisan [6], which has quadratic stretch and can be computed by bilinear functions. We show that composing it with itself, which is a take on an idea of Naor and Naor [7], yields the desired generators.

We first describe a generic composition lemma, which refers to generators of bounded locality, and then set-up a simple iterative process that yields generators of increased stretch (and larger locality). Next, we present versions of these ingredients that refer to generators that can be computed by polynomials of bounded degree. Lastly, we adapt the latter to support multilinear computation.

The notations used in this appendix are different from those used in the main text. The main parameter is the seed length, denoted k, and throughout this appendix the stretch and bias will be stated as functions of k. However, for the final application, given $t' \in \mathbb{N}$, we shall set $t'' = \mathrm{poly}(t')$ and $n = k/(t''-2)$, and obtain a $(t''-2)$-linear generator that outputs sequences of length $n^{t'+2}$ with bias at most 2^{-n}, where n is as in the main text.

Notation. We shall extensively use the notation U_m, which represents a random variable uniformly distributed over $\{0,1\}^m \equiv \mathrm{GF}(2)^m$. Throughout the text, we assume that the stretch function is super-linear and monotonically increasing, and that the bias bound in monotonically non-increasing. We recall the following standard definition.

Definition A.1 (generators of bounded bias w.r.t low-degree tests): *We say that* $G : \{0,1\}^k \to \{0,1\}^{s(k)}$ *has* bias at most $\epsilon(k)$ with respect to tests of degree d *if for every polynomial* (test) $T : \{0,1\}^{s(k)} \to \{0,1\}$ *of degree* d *it holds that*

$$\frac{1}{2} \cdot \left| \mathrm{E}\left[(-1)^{T(G(U_k))}\right] - \mathrm{E}\left[(-1)^{T(U_{s(k)})}\right] \right| \;\leq\; \epsilon(k). \tag{39}$$

The function $s:\mathbb{N}\to\mathbb{N}$ *is called the* stretch *of* G*, and* $\epsilon:\mathbb{N}\to[0,1]$ *is called its* bias.

The l.h.s. of Eq. (39) equals the total variation distance between the "verdict" of T in the two cases (i.e., the statistical difference between $T(G(U_k))$ and $T(U_{s(k)})$). Indeed, ϵ-bias generators (cf., [7] and [1, Sec. 8.5.2]) correspond to the special case of linear tests (i.e., $d = 1$). In general, whenever we talk of bias without specifying the degree, we mean bias with respect to linear tests.

A.1 A general composition lemma

Although we are interested in small-bias generators (i.e., bias w.r.t linear tests), it will be useful to have the following composition result that refers to generators with respect to tests of bounded degree. Specifically, we consider generators of *bounded locality* and specified stretch, which have small bias with respect to polynomials of bounded degree. Recall that a function is said to have locality ℓ if each bit in its output is a function of at most ℓ bits in its input (cf. [6]).

Lemma A.2 (composition of generators, a special case): *For* $i \in \{1,2\}$*, let* d_i *and* ℓ_i *be constants,* $s_i : \mathbb{N} \to \mathbb{N}$ *be a stretch function, and* $\epsilon_i : \mathbb{N} \to (0,1]$ *be a bias bound. Suppose that* $G_i : \{0,1\}^k \to \{0,1\}^{s_i(k)}$ *has locality* ℓ_i *and bias at most* $\epsilon_i(k)$ *with respect to all tests* (i.e., polynomials) *of degree* d_i*. Then,* $G = G_2 \circ G_1 : \{0,1\}^k \to \{0,1\}^{s_2(s_1(k))}$ *has locality* $\ell_2 \cdot \ell_1$ *and bias at most* $\epsilon(k) = \epsilon_1(k) + \epsilon_2(s_1(k))$ *with respect to all tests of degree* $d = \min(d_2, d_1/\ell_2)$*.*

Hence, the degree of tests that the composed generator withstands is the minimum between the degree withstand by the outer generator and a $1/\ell_2$ fraction of the degree withstand by the inner generator, where ℓ_2 is the locality of the outer generator. A more general statement allows the degree d_i and the locality ℓ_i to be functions of k; in this case, G has bias at most $\epsilon(k) = \epsilon_1(k) + \epsilon_2(s_1(k))$ with respect to all tests of degree $d(k) = \min(d_2(s_1(k)), d_1(k)/\ell_2(s_1(k)))$.

Proof: We use a hybrid argument (cf. [1, Sec. 8.2.3.3]), while considering the following three distributions:

1. The pseudorandom output $G(U_k) = G_2(G_1(U_k))$.
2. The intermediate hybrid $G_2(U_{s_1(k)})$.
3. The uniform distribution $U_{s_2(s_1(k))}$.

Let T be an arbitrary test of degree d. Then,

$$\begin{aligned}
&\left|\mathrm{E}\left[(-1)^{T(G_2(G_1(U_k)))}\right] - \mathrm{E}\left[(-1)^{T(G_2(U_{s_1(k)}))}\right]\right| \\
&\quad = \left|\mathrm{E}\left[(-1)^{T'(G_1(U_k))}\right] - \mathrm{E}\left[(-1)^{T'(U_{s_1(k)})}\right]\right| \\
&\quad \leq \epsilon_1(k),
\end{aligned}$$

since $T' = T \circ G_2$ is a test of degree $d \cdot \ell_2 \leq d_1$ (and G_1 has bounded bias w.r.t such tests). Using the fact that $d \leq d_2$ (and the hypothesis regarding G_2), we have

$$\left|\mathrm{E}\left[(-1)^{T(G_2(U_{s_1(k)}))}\right] - \mathrm{E}\left[(-1)^{T(U_{s_2(s_1(k))})}\right]\right| \leq \epsilon_2(s_1(k)).$$

Combining both bounds, the claim follows. ■

A.2 An iterative construction

The basic idea is to iteratively compose a small bias generator of constant locality, which fools linear tests, with itself. But for the process to work we need the inner generator, in the composition, to fool tests with degree that at least equals the locality of the outer generator (see Lemma A.2). Using Viola's result [11], we can get such an inner generator by taking the sum of a constant number of instances of the current generator (which fool linear tests), and keep using the original generator as the outer generator. Details follow.

The starting point. Let G be a small bias generator that has constant locality ℓ, some stretch $s : \mathbb{N} \to \mathbb{N}$, and bias $\epsilon : \mathbb{N} \to (0, 1]$ with respect to linear tests. We shall use G as the outer generator in all compositions. In addition, we shall use G as the inner generator in the first iteration; that is, we let $G^{(0)}$ equal G; hence, $G^{(0)}$ has locality $\ell^{(0)} = \ell$, stretch $s^{(0)}(k) = s(k)$ and bias at most $\epsilon^{(0)}(k) = \epsilon(k)$.

Iteration $i \in \mathbb{N}$**.** Given an $\ell^{(i-1)}$-local generator $G^{(i-1)} : \{0,1\}^k \to \{0,1\}^{s^{(i-1)}(k)}$ that fools linear tests with bias $\epsilon^{(i-1)}(k)$, we first obtain a generator $\widehat{G}^{(i-1)} : \{0,1\}^{\ell \cdot k} \to \{0,1\}^{s^{(i-1)}(k)}$ that fools degree ℓ tests with bias $O(\epsilon^{(i-1)}(k)^{2^{-(\ell-1)}})$, by XORing ℓ instances of $G^{(i-1)}$ (see Viola [11]); that is, $\widehat{G}^{(i-1)}(x_1, ..., x_\ell) = \bigoplus_{j \in [\ell]} G^{(i-1)}(x_j)$. Hence, $\widehat{G}^{(i-1)}$ has locality $\ell' = \ell \cdot \ell^{(i-1)}$, stretch $s'(k) = s^{(i-1)}(k/\ell)$, and bias $\epsilon'(k) = O(\epsilon^{(i-1)}(k/\ell)^{2^{-(\ell-1)}})$ w.r.t tests of degree ℓ. Next, applying Lemma A.2, we obtain $G^{(i)} = G \circ \widehat{G}^{(i-1)}$, and observe that $G^{(i)}$ has locality $\ell^{(i)} = \ell \cdot \ell' = \ell^2 \cdot \ell^{(i-1)}$, stretch $s^{(i)}(k) = s(s'(k)) = s(s^{(i-1)}(k/\ell))$, and bias at most $\epsilon^{(i)}(k) = \epsilon(s(k)) + \epsilon'(k) = O(\epsilon^{(i-1)}(k/\ell)^{2^{-\ell}})$ with respect to linear tests (since the locality of G equals the degree that $\widehat{G}^{(i-1)}$ is guaranteed to fool).

Hence, after $\tau \in \mathbb{N}$ iterations, we obtain a generator (i.e., $G^{(\tau)}$) that has locality $\ell^{(\tau)} = \ell^{2\tau} \cdot \ell^{(0)} = \ell^{2\tau+1}$, bias at most $\epsilon^{(\tau)}(k) < O(1)^\tau \cdot \epsilon(k/\ell^\tau)^{(2^{-\ell})^\tau}$ with respect to linear tests, and stretch $s^{(\tau)}(k) = s^{\circ^{\tau+1}}(k/\ell^\tau)$, where s°^τ} denotes s composed with itself τ times (i.e., $s^{\circ^i}(k) = s(s^{\circ^{i-1}}(k))$ and $s^{\circ^1}(k) = s(k)$). Hence, we obtain the following result, which is not used in this work (and is stated merely for sake of future reference).

Corollary A.3 (amplifying the stretch of small-bias generators of bounded locality): *Let G be a generator that has constant locality ℓ, stretch $s : \mathbb{N} \to \mathbb{N}$, and bias $\epsilon : \mathbb{N} \to (0,1]$ with respect to linear tests. Suppose that $\tau \leq 0.5 \log_\ell k$ and that $s(k) \geq k^\alpha$ for some constant $\alpha > 1$. Then, $G^{(\tau)}$ has locality $\ell^{(\tau)} = \ell^{2\tau+1}$, stretch $s^{(\tau)}(k) \geq k^{\alpha^{\tau+1}/2}$, and bias at most $\epsilon^{(\tau)}(k) < O(1)^\tau \cdot \epsilon(\sqrt{k})^{(2^\tau)^{-\ell}}$ with respect to linear tests.*

A.3 Adaptation to constructions of bounded degree generators

We actually seek a construction of generators that can be computed by bounded degree polynomials rather than by functions of bounded locality. The foregoing analysis extends in a straightforward manner to the current case, yielding the following.

Lemma A.4 (Lemma A.2, revisited): *For $i \in \{1,2\}$, let d_i and D_i be constants, $s_i : \mathbb{N} \to \mathbb{N}$ be a stretch function, and $\epsilon_i : \mathbb{N} \to (0,1]$ be a bias bound. Suppose that $G_i : \{0,1\}^k \to \{0,1\}^{s_i(k)}$ can be computed by a sequence of polynomials of degree D_i and has bias at most $\epsilon_i(k)$ with respect to all tests of degree d_i. Then, $G = G_2 \circ G_1 : \{0,1\}^k \to \{0,1\}^{s_2(s_1(k))}$ can be computed by a sequence of polynomials of degree $D_2 \cdot D_1$ and has bias at most $\epsilon(k) = \epsilon_1(k) + \epsilon_2(s_1(k))$ with respect to all tests of degree $d = \min(d_2, d_1/D_2)$.*

The proof is identical to the proof of Lemma A.2, and the iterative construction works as well.

Corollary. We obtain a concrete result that is analogous to Corollary A.3 by using the generator of Mossel, Shpilka, and Trevisan [6], which has $s(k) = \Omega(k^2)$ and $\epsilon(k) = 2^{\Omega(k)}$, with $D = 2$. Observe that, after $\tau \in \mathbb{N}$ iterations, we obtain a generator (i.e., $G^{(\tau)}$) that can be computed by a sequence of polynomials of degree $D^{(\tau)} = D^{2\tau+1} = 2^{2\tau+1}$, has stretch $s^{(\tau)}(k) = s^{\circ^{\tau+1}}(k/D^t) = \Omega(k/2^\tau)^{2^{\tau+1}}$, and bias at most $\epsilon^{(\tau)}(k) = O(1)^\tau \cdot \epsilon(k/D^\tau)^{(2^{-D})^\tau} = \exp(\exp(-O(\tau)) \cdot k)$. Hence, seeking stretch of the form n^σ, we set $\tau = \log_2 \sigma$, and obtain degree $2 \cdot \sigma^2$ and bias at most $2^{-\mathrm{poly}(1/\sigma) \cdot k}$.

A.4 Adaptation to multilinear constructions of bounded degree

Actually, we need the construction to be multilinear; that is, we seek a construction of generators that can be computed by bounded degree multi-linear functions. While the transformation from

fooling linear tests to fooling tests of constant degree preserves the multilinearity of the generator (since it XORs independently generated outputs of the original generator), the composition lemma does not necessarily preserve multilinearity. That is, even if both G_i's are computed by sequences of multilinear functions (of bounded degree), their composition may not be so (since G_2 may multiply output bits of G_1 that depend on the same variable-block of inputs in the seed of G_1). Still, a small modification suffices to provide multilinearity.

Lemma A.5 (Lemma A.4, revisited): *For $i \in \{1,2\}$, let d_i and D_i be constants, $s_i : \mathbb{N} \to \mathbb{N}$ be a stretch function, and $\epsilon_i : \mathbb{N} \to (0,1]$ be a bias bound. Suppose that $G_i : \{0,1\}^k \to \{0,1\}^{s_i(k)}$ can be computed by a sequence of D_i-linear functions, where an m-linear function from $\mathrm{GF}(2)^k$ to $\mathrm{GF}(2)$ is linear in each of the m* (equal-length) *blocks of variables, and has bias at most $\epsilon_i(k)$ with respect to all tests of degree d_i. Let $G'_1 : \{0,1\}^k \to \{0,1\}^{D_2 \cdot s_1(k/D_2)}$ be an algorithm that partitions its input to D_2 equal-length parts, applies G_1 to each part, and concatenate the results. Then, $G = G_2 \circ G'_1 : \{0,1\}^k \to \{0,1\}^{s_2(D_2 \cdot s_1(k/D_2))}$ can be computed by a sequence of $D_2 \cdot D_1$-linear functions and has bias at most $\epsilon(k) = D_2 \cdot \epsilon_1(k/D_2) + \epsilon_2(s_1(k))$ with respect to all tests of degree $d = \min(d_2, d_1/D_2)$.*

Proof: We first observe that, by construction, G is $D_2 \cdot D_1$-linear, since the different D_2 (equal length) blocks of the input to G_2 depend on disjoint (k/D_2)-bit long parts of the seed of G'_1 (i.e., the i^{th} block in the input to G_2 depends on the i^{th} part of the seed of G'_1). Specifically, on input $\overline{x} = (x_1, ..., x_{D_2}) \in \{0,1\}^{D_2 \cdot (k/D_2)}$, each monomial in the computation of $G(\overline{x})$ depends on at most D_2 bits of $G'_1(\overline{x})$, which by the D_2-linearity of G_2 occur in different parts in $G'_1(\overline{x}) = (G_1(x_1), ..., G_1(x_{D_2}))$, whereas each of these parts is computed by a D_1-linear function of the corresponding seed (i.e., the i^{th} part of $G'_1(\overline{x})$ appears in $G_1(x_i)$, and is computed by a D_1-linear function of x_i).

All that remains is to observe that G'_1 essentially inherits the bias bound of G_1; specifically, we show that G'_1 has bias at most $D_2 \cdot \epsilon_1(k/D_2)$ with respect to tests of degree d_1. This is shown using a hybrid argument, where the i^{th} hybrid, denoted H_i, consists of i independent copies of $G_1(U_{k/D_2})$ followed by $D_2 - i$ independent copies of $U_{s_1(k/D_2)}$. (Indeed, $H_0 = U_{D_2 \cdot s_1(k/D_2)}$ and $H_{D_2} = G'_1(U_k)$.) Then, for any test T of degree d_1, it holds that

$$\begin{aligned}
&\left|\mathrm{E}\left[(-1)^{T(U_{D_2 \cdot s_1(k/D_2)}))}\right] - \mathrm{E}\left[(-1)^{T(G'_1(U_k))}\right]\right| \\
&\quad \leq \sum_{i \in [D_2]} \left|\mathrm{E}\left[(-1)^{T(H_{i-1})}\right] - \mathrm{E}\left[(-1)^{T(H_i)}\right]\right| \\
&\quad = \sum_{i \in [D_2]} \left|\mathrm{E}\left[(-1)^{T_i(U_{s_1(k/D_2)}))}\right] - \mathrm{E}\left[(-1)^{T_i(G_1(U_{k/D_2}))}\right]\right| \\
&\quad \leq D_2 \cdot \epsilon_1(n/D_2),
\end{aligned}$$

where $T_i(z)$, which may be viewed as a distribution over tests of degree d_1, generates $i-1$ independent copies of $G_1(U_{k/D_2})$, denoted $v_1, ..., v_{i-1}$, and $D_2 - i$ independent copies of $U_{s_1(k/D_2)}$, denoted $u_1, ..., u_{D_2-i}$, and returns $T(v_1, ..., v_{i-1}, z, u_1, ..., u_{D_2-i})$. Indeed, we use the fact that H_{i-1} and H_i differ only in their i^{th} part (which is $U_{s_1(k/D_2)}$ in H_{i-1} and $G_1(U_{k/D_2})$ in H_i). ■

Conclusion. Starting with the generator of Mossel, Shpilka, and Trevisan [6], while noting that it is actually bilinear, and using the iterative construction of Section A.2 with the composition of Lemma A.5, we get the desired generator (which is stated in the terms used in the main text).

Theorem A.6 (a small-bias generator of arbitrary large polynomial stretch that can be computed by $O(1)$-linear functions): *For every $t' \in \mathbb{N}$, there exist t'' and an explicit construction of an $(t''-2)$-linear generator $G_{\mathtt{sb}} : \mathrm{GF}(2)^{(t''-2)\cdot n} \to \{0,1\}^{n^{t'+2}}$ has bias at most 2^{-n}. Furthermore, $t'' = O(t')^2$.*

Given all the foregoing, the following proof is straightforward. It is being detailed here for sake of tedious verification.

Proof: We just mimic the argument of Section A.2, while using the generator of [6] as our "pivot" generator, and using the composition result of Lemma A.5 in all iterations. Specifically, letting $\epsilon(k) = 2^{\Omega(k)}$, we start with an ϵ-bias generator of stretch $s(k) = \Omega(k^2)$ that is computed by bilinear function. Recall that we let $G^{(0)}$ equal G, and so $D^{(0)} = D = 2$, $s^{(0)}(k) = s(k)$ and $\epsilon^{(0)}(k) = \epsilon(k)$.

Next, given a $D^{(i-1)}$-linear generator $G^{(i-1)} : \{0,1\}^k \to \{0,1\}^{s^{(i-1)}(k)}$ that fools linear tests with bias $\epsilon^{(i-1)}(k)$, we first obtain a $2 \cdot D^{(i-1)}$-linear generator $\widehat{G}^{(i-1)} : \{0,1\}^{2\cdot k} \to \{0,1\}^{s^{(i-1)}(k)}$ that fools quadratic tests with bias $O(\epsilon^{(i-1)}(k)^{1/2})$, by XORing two instances of $G^{(i-1)}$. Hence, $\widehat{G}^{(i-1)}$ has stretch $s'(k) = 2 \cdot s^{(i-1)}(k/2)$, and bias $\epsilon'(k) = O(\epsilon^{(i-1)}(k/2)^{1/2})$ w.r.t quadratic tests.

Next, applying Lemma A.5, we obtain $G^{(i)} = G \circ \widetilde{G}^{(i-1)}$, where $\widetilde{G}^{(i-1)}$ is obtained from $\widehat{G}^{(i-1)}$ by partitioning the seed into two (equal-length) parts and applying $\widehat{G}^{(i-1)}$ on each part. Hence, $G^{(i)}$ is $D^{(i)}$-linear, for $D^{(i)} = 2 \cdot 2D^{(i-1)} = 2^{2i+1}$, and has stretch $s^{(i)}(k) = s(s'(k)) = s(2 \cdot s^{(i-1)}(k/2))$, and bias at most $\epsilon^{(i)}(k) = \epsilon(s(k)) + \epsilon'(k) = O(\epsilon^{(i-1)}(k/2)^{1/2})$ with respect to linear tests. Assuming that $s(k) \geq c \cdot k^2$, for some constant $c > 0$, we have

$$\begin{aligned} s^{(i)}(k) &\geq c \cdot (2 \cdot s^{(i-1)}(k/2))^2 \\ &= (4c)^{2^{i+1}-1} \cdot s^{(0)}(k/2^i)^{2^i} \\ &= (4c)^{2^{i+1}-1} \cdot c \cdot (k/2^i)^{2^{i+1}} \\ &= \exp(-\widetilde{O}(2^i)) \cdot k^{2^{i+1}} \end{aligned}$$

and

$$\begin{aligned} \epsilon^{(i)}(k) &= O(\epsilon^{(i-1)}(k/2)^{1/2}) \\ &= O(1) \cdot \epsilon^{(0)}(k/2^i)^{2^{-i}} \\ &= \exp(\Omega(4^{-i} \cdot k)). \end{aligned}$$

Hence, after $\tau \in \mathbb{N}$ iterations, we obtain a $2^{2\tau+1}$-linear generator (i.e., $G^{(\tau)}$) that has stretch $s^{(\tau)}(k) > \exp(-\widetilde{O}(2^\tau)) \cdot k^{2^{\tau+1}}$ and bias at most $\epsilon^{(\tau)}(k) < \exp(\Omega(4^{-\tau} \cdot k))$. Thus, seeking stretch of the form k^σ, we set $\tau = \log_2 \sigma$, and obtain a $2 \cdot \sigma^2$-linear generator with stretch $\exp(-\widetilde{O}(\sigma)) \cdot k^{\sigma^2}$ and bias at most $2^{-\Omega(1/\sigma)^2 \cdot k}$. Letting $n = k/O(\sigma^2)$ (and using a finer partition of the k-bit long input seed), we can view this generator as being (k/n)-linear and having bias at most 2^{-n}. The theorem follows. ■

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il