Improved bounds on the AN-complexity of multilinear functions

Oded Goldreich*

January 10, 2020

Abstract

We consider arithmetic circuits with arbitrary large (multi-linear) gates for computing multi-linear functions. 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 were introduced by Goldreich and Wigderson (*ECCC*, TR13-043, 2013), and is called the AN-complexity.

The AN-complexity of a multi-linear function yields an upper bound on the size of depth-three Boolean circuits for computing the function, and it is not clear whether or not significantly smaller depth-three Boolean functions exist. 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 multi-linear function is a essential step towards proving that depth-three Boolean circuits for these functions requires exponential size.

In this work we present explicit multi-linear functions that require depth-two multi-linear 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 circuit (with general multi-linear gates) that computes f must use gates of arity at least $n^{1-\epsilon}$. 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 multi-linear 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.

Keywords: Depth-three Boolean circuits, arithmetic circuits, multi-linear circuits, multi-linear functions, random restrictions, small-bias generators.

*Department of Computer Science, Weizmann Institute of Science, Rehovot, Israel. Work done while visiting the Computer Science Department of Columbia University. E-mail: oded.goldreich@weizmann.ac.il

Contents

1 Introduction **1**
1.1 From canonical Boolean circuits to AN-complexity of multilinear circuits 1
1.2 Prior results regarding the AN-complexity of multi-linear functions 2
1.3 Our results 3
1.4 Outline of the proof of Theorem 1.7 3
1.4.1 The random restriction and its affect on gates with large blocks 4
1.4.2 Gates with small blocks: A special case 4
1.4.3 Gates with small blocks: The general case 5

2 Proof of Theorem 1.7 **6**
2.1 The form of a depth-two multilinear circuit that computes F 7
2.2 Three types of gates and how they will be handled 8
2.3 The actual handling of the three types 9
2.4 Wrapping-up and reaching a contradiction 13

3 Conclusions **16**

Acknowledgements **18**

References **18**

Appendix: On small-bias generators of large stretch **19**
A.1 A general composition lemma 19
A.2 An iterative construction 20
A.3 Adaptation to constructions of bounded degree generators 21
A.4 Adaptation to multilinear constructions of bounded degree 21

1 Introduction

Providing exponential lower bounds on the size of constant-depth Boolean circuits computing 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] suggested to consider some explicit multi-linear functions (as potential candidates for functions that require exponential-size depth-three circuits), and to start by establishing a size lower bound in a restricted model of ("canonical") depth-three circuits. Needless to say, the latter model covers the standard construction of depth-three Boolean circuits for parity, and seems quite natural (if not optimal) in the context of computing multi-linear functions.

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

The restricted model of depth-three canonical Boolean circuits is closely related to a model of multi-linear arithmetic circuits with general multi-linear gates, and the complexity measure that is related to the size of the former canonical depth-three Boolean circuits is the maximum between the arity and number of gates in the multi-linear circuits. A few clarifications are in place.

- By multi-linear functions we mean functions of the form $f : \{0,1\}^{t \cdot n} \to \{0,1\}$ such that

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

  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 such that each monomial takes a single variable from each of the t blocks of variables. We often relate to such function as t-linear.

  The t-ary array $(f_{i_1,...,i_t})_{i_1,...,i_t \in [n]}$ is the tensor corresponding to f. The tensor indicate which of the possible n^t monomials are actually included in the function.

- An arithmetic circuit with arbitrary gates that computes a multi-linear function is called multilinear if each of the monomials computed by each of its gates multiplies the results of sub-circuits that depend on *disjoint* blocks of variables. Beyond this global (to the circuit) syntactic requirement, the gates may be arbitrary.

Clearly, any multi-linear function $f : \{0,1\}^{t \cdot n} \to \{0,1\}$ can be computed by a multi-linear 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 1.1 (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).

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).) Applying a Valiant-like idea [9], which can be actually traced to the reduction of Circuit-SAT to SAT, Goldreich and Wigderson [4] showed the following.

Theorem 1.2 (The size of depth-three Boolean circuits is at most exponential in the AN-complexity [4]): *Any multilinear function f can be computed by a depth-three Boolean circuit of size* $\exp(\mathtt{AN}(f))$.

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 t-linear function $f : \{0,1\}^{tn} \to \{0,1\}$ requires proving that $\mathtt{AN}(f) = \omega(n^{1/2})$.

1.2 Prior 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 $\mathtt{AN}(\mathtt{PAR}_n) = \Omega(n^{1/2})$. Our interest is in $O(1)$-linear functions that have AN-complexity $\omega(n^{1/2})$. (Note that such a result means that such a function does not have a *canonical* depth-three Boolean circuit of size $\exp(O(n^{1/2}))$.) We start with an upper bound that sets the limit on such lower bounds.

Theorem 1.3 (a generic upper bound [4]): *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 $O(n^{2/3})$. 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 1.4 (existential lower bound [4]): *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 $\Omega(n^{2/3})$. Of course, the goal is obtaining lower bounds for explicit functions (and Theorems 1.3 and 1.4 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. [8]), which was established by Goldreich and Wigderson [4].

Theorem 1.5 ($\omega(n^{1/2})$ lower bounds for explicit functions [3]):

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 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 $g_{i,j}(r,s)$ is a bilinear form that describes a bit in the output of a small bias generator (see Definition A.1).

1.3 Our results

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

Open Problem 1.6 ($\omega(n^{2/3})$ 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* $\mathtt{AN}_2(f) = \omega(n^{2/3})$.

Our result resolves the second problem. Specifically, we prove the following result.

Theorem 1.7 (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 casew 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].

1.4 Outline of the proof of Theorem 1.7

Let us take a look at a generic multilinear circuit of depth two and AN-complexity m that computes f. 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). 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.

It is instructive to consider the blocks of variables that feed each of the intermediate gates. We denote by $B_i \subseteq [t]$ the set of variable-blocks that feed G_i, and note that 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 $[t] \setminus B_i$.

1.4.1 The random restriction and its affect on gates with large blocks

Now, 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 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 is (randomly) restricted to a bilinear function $f_{i_1,...,i_{t-2}}$ such that

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

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

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. 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., all $f_{i_1,...,i_{t-2}}$ are not 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 1.7 would follow (when we pick the best depth-two circuit, so that $m = \mathtt{AN}_2(f)$).

1.4.2 Gates with small 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 a subset of $\{t-1,t\}$ or a subset of $[t-2]$. Under this (unjustified) assumption, the circuit C has the form

$$\begin{aligned} C(x^{(1)},...,x^{(t-2)},y,z) &= \sum_{j\in J} F_j(x^{(1)},...,x^{(t-2)}) \cdot G_j(y,z) &(2)\\ &+ \sum_{(j_1,j_2)\in K} F_{j_1,j_2}(x^{(1)},...,x^{(t-2)}) \cdot G_{j_1}(y) \cdot G_{j_2}(z), &(3)\end{aligned}$$

where the F_j's and F_{j_1,j_2}'s are $(t-2)$-linear functions, and $J \subseteq [m]$ and $K \subset [m]^2$. Recalling that, in order to get rid of the larger B_i's, we picked a random assignment to the variables in the block 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 necasserily the F_{j_1,j_2}'s)[1] evaluate to 0, then the bilinear function computed by the residual circuit corresponds to a matrix of rank at most m (since each $G_{j_1}(y) \cdot \sum_{j_2:(j_1,j_2)\in K} G_{j_2}(z)$ corresponds to a matrix of rank at most 1).[2] 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 1.7 would follow, provided that we select an adequate function f (which is quite easy to do).

[1] Footnote 3 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).

[2] In general, each bilinear function that is a product of two linear functions corresponds to a matrix of rank 1 (unless it is identically zero). This correspondence is the pivot of the connection between the AN-complexity of bilinear functions and matrix rigidity [4], which in turn is the starting point of [3].

Of course, there is no reason to believe that we may be so extremely lucky (and have all F_j's evaluate to 0). What we do in this case is selected $m+1$ random assignments to the variables of the blocks in $[t-2]$, and consider the $m+1$ vectors of values of the F_j's (i.e., the i^{th} vector describes the values of all F_j's under the i^{th} assignment).[3] Now, we select a non-zero linear combination of these vectors that sums-up to zero, and look at the corresponding linear combinations of the computations of C and the evaluations of f.

The key observation is that the linear combination of the computations of (the restricted) circuit C yields a bilinear function that corresponds to a matrix of rank m, since the contribution of the $\sum_{j\in J} F_j G_j$ cancels out (because, for each $j \in J$, the linear combination of the assignments to F_j evaluates to 0). Hence, if the corresponding linear combinations of the evaluations of (the restricted) function f yields a bilinear function of higher rank, then we reach a contradiction again. For this to happen, it suffices that *each linear combinations of slices of f corresponds to a matrix of rank higher than m*, where a slice of f is a bilinear function $f_{i_1,...,i_{t-2}}$ (as in Eq. (1)). 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 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 $m+1$ 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 $I_1 \times \cdots \times I_{t-2}$. Hence, 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 variable that feeds this gate. Observing that this event occurs with probability at least $1-(b\cdot(m/n))^{|B_i\cap[t-2]|}$. Hence, we reach a contradiction unless $m\cdot((b\cdot m)/n)^d \geq 1$, which implies $m > n^{\frac{t-2}{t-1}\cdot\frac{d}{d+1}} > n^{1-\epsilon}$, provided that $\min(d+1, t-2) \geq 2/\epsilon$.

1.4.3 Gates with small 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 $1,...,d' < d$. Then, the contribution of this gate to the computation of C has the form

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

where F_i is an arbitrary $(t-2-d')$-linear function (analogously to Eq. (2)). This means that for the sets $I_1,...,I_{t-2}$ selected as above, we have to find a set $S \subseteq I_1 \times \cdots \times I_{t-2}$ such that for every

[3] We can afford to consider all $2^{m+1}-1$ linear combinations, but we can not afford to consider $2^{\Omega(m^2)}$ linear combination, because we aim at $m = \omega(n^{1/2})$. This is the reason that this argument is applied to the F_j's, but not to the F_{j_1,j_2}'s.

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 $(m+1)^{1/(t-2)}$). 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 $b^d \cdot m < n/2$).[4]

So we are left with the case of small B_i's that intersect $\{t-1, t\}$ at a single point. For simplicity consider $B_i = \{1, ..., d', t\}$ such that $d' < d$. The crucial observation here is that the expected number of variables from blocks $[d']$ that will be set to 1 by any of the random assignments is $(m+1)^{d'/(t-2-d)} \cdot (m/n)^{d'} < ((m+1)^{1+\frac{1}{t-2+d}}/n)^{d'} < 1$. Hence, the contribution of such a gate to the rank of the matrix that corresponds to the bilinear function computed by the residual circuit is actually maximized in the case $d' = 0$, which was considered above.

This completes the rough sketch of the proof of Theorem 1.7, 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 corresponds to a matrix of high rank.

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 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. (4), coming next). Specifically, we shall use a small bias pseudorandom generator that can be evaluated by a multi-linear function. In particular, we use an $\exp(-\Omega(n))$-bias generator that stretches $\mathrm{poly}(t) \cdot n$ bits into n^t bits 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.

2 Proof of Theorem 1.7

We start with an explicit presentation of the multi-linear functions that we shall analyze. For t' and t'' to be determined, we consider the $(t'+t'')$-linear function $F : \mathrm{GF}(2)^{(t'+t'')\cdot n} \to \mathrm{GF}(2)$ defined as

$$F(x^{(1)}, x^{(2)}, ..., x^{(t'+t'')}) = \sum_{i_1,...,i_{t'+2}\in[n]} G_{\mathtt{sb}}(x^{(t'+3)}, ..., x^{(t'+t'')})_{(i_1,...,i_{t'+2})} \cdot \prod_{j\in[t'+2]} x^{(j)}_{i_j} \qquad (4)$$

[4] The added condition is used in the analysis of the psedorandom 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 $m = n^{1-\epsilon}$.

where $G_{\mathtt{sb}} : \mathrm{GF}(2)^{(t''-2)\cdot n} \to \mathrm{GF}(2)^{n^{t'+2}}$ is an $\exp(-\Omega(n))$-bias generator that is computed by $(t''-2)$-linear functions. Specifically, the $(i_1, ..., i_{t'+2})^{\text{th}}$ bit in the output of $G_{\mathtt{sb}}$, denoted $G_{\mathtt{sb}}(x^{(t'+3)}, ..., x^{(t'+t'')})_{(i_1,...,i}$ is computed by a $(t''-2)$-linear function.

Intuitively, the first $t'+2$ coordinates of the tensor that describes the function computed by F correspond to a pseudorandom (i.e., small-bias) tensor, where the pseudorandomness is provided by the last $t''-2$ coordinates. We shall consider the $n^{t'}$ two-dimensional slices of the former $(t'+2)$-dimensional tensor, where the slices are aligned with coordinates $t'+1$ and $t'+2$. That is, fixing an arbitrary assignment $s \in \{0,1\}^{(t''-2)n}$ to the last $t''-2$ blocks of variables (i.e., fixing a seed to the small-bias generator), the slice $(i_1, ..., i_{t'}) \in [n]^{t'}$ of the (the tensor of the) residual function $F_s(\cdots) = F(\cdots, s)$ is the 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 $G_{\mathtt{sb}}(s)_{(i_1,...,i_{t'},j_1,j_2)}$.

We shall need to pick a large enough t' for the analysis of these slices, and this will require setting $t''-2 = \Omega(t')$ so that the $n^{t'+2}$-long sample space can have small bias.

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 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. That is, 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 indices of the variable-blocks on which G_i depends, and we say that G_i is $\mathtt{BL}(i)$-linear. 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

$$F = \sum_{I\in\mathcal{C}} c_I \cdot \prod_{i\in I} G_i \tag{5}$$

$$= \sum_{i\in[m]:\mathtt{BL}(i)\ni t'+1} G_i \cdot \sum_{I\in\mathcal{C}:I\ni i} c_I \cdot \prod_{j\in(I\setminus\{i\})} G_j, \tag{6}$$

where each G_i has arity at most m (and an empty product is defined as identical to 1 (and $\prod_{j\in\emptyset} G_j = 1$)).[5] Hence, each G_i depends on at most m variables in each of the blocks in the corresponding $\mathtt{BL}(i)$. Letting

$$F_i \stackrel{\text{def}}{=} \sum_{I\in\mathcal{C}:I\ni i} c_I \cdot \prod_{j\in(I\setminus\{i\})} G_j, \tag{7}$$

we get

$$F = \sum_{i\in[m]:\mathtt{BL}(i)\ni t'+1} G_i \cdot F_i \tag{8}$$

[5] Indeed, for each monomial in Eq. (5), we focus on 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.

where we treat F_i as arbitrary $([t] \setminus \mathtt{BL}(i))$-linear functions.

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 corresponds to a matrix of rank $O(m)$, where this bilinear is derived from the r.h.s. of Eq. (8). In contrast, considering the l.h.s. of Eq. (8), we shall later show that this is unlikely to happen for a random seed, unless $m = \Omega(n^{1-\epsilon})$. Here we focus on the former part (which was sketched in Section 1.4).

Towards this end, we fix such a seed $s \in \{0,1\}^{(t''-2)\cdot n}$, 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 monomials in Eq. (5) (equiv., the G_i's in Eq. (8)) to three types:

1. Monomials in which variables from blocks $t'+1$ and $t'+2$ are fed to different auxiliary functions; equivalently, functions G_i such that $t'+2 \notin \mathtt{BL}(i)$, which means that the variables of block $t'+2$ feed the corresponding function F_i (of Eq. (8)).[6]

   Such monomials (resp., functions) contribute to the rank of a matrix corresponding to a related bilinear function (which depends on variables in blocks $t'+1$ and $t'+2$), and 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, and otherwise we will bound the contribution of G_i in expectation by using the fact that it gets simplified by a random restriction (of the type discussed in Section 1.4).

2. Monomials in which variables from blocks $t'+1$ and $t'+2$ are fed to the same auxiliary function along with variables from *many* blocks in $[t']$; equivalently, functions G_i for which $t'+1, t'+2 \in \mathtt{BL}(i)$ and $|\mathtt{BL}(i) \cap [t']| \geq d$.

   Such monomials (resp., functions) contribute to the non-zero entries of a matrix corresponding to a related bilinear function, and this contribution will be bounded by using d blocks in $[t']$. Specifically, we will bound the contribution of G_i in expectation by using the fact that it gets simplified by a random restriction.

3. Monomials in which variables from blocks $t'+1$ and $t'+2$ are fed to the same auxiliary function along with variables from *few* blocks in $[t']$; equivalently, functions G_i for which $t'+1, t'+2 \in \mathtt{BL}(i)$ and $|\mathtt{BL}(i)| < d$.

   We shall show that such monomials (resp., functions) can be ignore viz a viz the aforementioned matrices. This will be done by taking a suitable linear combination of relatively few slices that correspond to different random restrictions of the circuit. Doing so will indeed complicate the analysis of the first case, 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.

A contradiction will follow by showing that the corresponding matrix in the tensor that is associated with F itself has higher rank than the upper bound established above. (A slightly better result can

[6] Recall that throughout this section, we focus on G_i's such that $t'+1 \in \mathtt{BL}(i)$.

be obtained by considering the rigidity of both matrices with respect to this rank (see Remark 2.3), but this is not really necessary.)

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 $[t'+2]$.)

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 resulting residual function $F_i \cdot G_i$ is a bilinear function in $x^{(t'+2)}$ and $x^{(t'+1)}$. Furthermore, the n-by-n matrix that corresponds to this residual bilinear function has rank at most 1 (since it is an outer product of two vectors (i.e., the vectors representing the residual 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 corresponds to the residual bilinear function $\sum_{i \in T_1} F_i G_i$ has rank at most $|T_1| \leq m$.

Foreseeing the treatment of Type 3, which was outlined in Section 1.4, we need to handle the sum of $F_i \cdot G_i$ taken over many random assignments. These 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. 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 $(I_1(k_1), ..., I_{t'}(k_{t'})) \in [b]^{t'}$, which in turn specifies a random assignment $(\mathtt{u}(I_1(k_1)), ..., \mathtt{u}(I_{t'}(k_{t'}))) \in \{0,1\}^{t' \cdot n}$.

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

$$\sum_{(k_1,...,k_{t'}) \in S} F_i(\mathtt{u}(I_1(k_1)), ..., \mathtt{u}(I_{t'}(k_{t'})), x^{(t'+2)}) \cdot G_i(\mathtt{u}(I_1(k_1)), ..., \mathtt{u}(I_{t'}(k_{t'})), x^{(t'+1)}) \tag{9}$$

to the rank of the corresponding matrix. 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 so we may replace G_i by the actual function, denoted G'_i, that this gate applies to variables in $\mathtt{BL}(i)$. (We could do the same for F_i, but there is no instructive benefit in doing so.) For sake of simplicity, assume that $\mathtt{BL}(i) = [d'] \cup \{t'+1\}$, for some $d' \geq 0$. Then, we can re-write Eq. (9) as follows

$$\begin{aligned} &\sum_{k_1,...,k_{d'} \in [b]} G'_i(\mathtt{u}(I_1(k_1)), ..., \mathtt{u}(I_{d'}(k_{d'})), x^{(t'+1)}) \\ &\qquad \cdot \sum_{k_{d'+1},...,k_{t'} \in [b]:(k_1,...,k_{t'}) \in S} F_i(\mathtt{u}(I_1(k_1)), ..., \mathtt{u}(I_{t'}(k_{t'})), x^{(t'+2)}) \\ &= \sum_{k_1,...,k_{d'} \in [b]} G'_i(\mathtt{u}(I_1(k_1)), ..., \mathtt{u}(I_{d'}(k_{d'})), x^{(t'+1)}) \cdot F_{i,S}^{(k_1,...,k_{d'})}(x^{(t'+2)}) \end{aligned} \tag{10}$$

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

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

with $S^{(k_1,...,k_{d'})} \stackrel{\text{def}}{=} \{(k_{d'+1},...,k_{t'}) \in [b]^{t'-d'} : (k_1,...,k_{t'}) \in S\}$. Looking at Eq. (10), 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 cancelation 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.[7] Hence, the expected number of terms in Eq. (10) that do not vanish is at most $b^{d'} \cdot (m/n)^{d'}$, which means that the expected rank of the matrix corresponding to Eq. (10) is at most $(b \cdot m/n)^{d'}$, where the expectations are taken uniformly over the choices of $\overline{I} \in \binom{[n]}{b}^{t'}$.

(Indeed, the fact that $S = S(\overline{I})$ is determined based on $\overline{I}$ is immaterial, since anyhow we consider all $b^{d'}$ terms, and for each term we only consider its vanishing due to $G'_i(\mathtt{u}(I_1(k_1)),...,\mathtt{u}(I_{d'}(k_{d'})),x^{(t'+1)}) \equiv 0$.) Note that $b < n/m$ implies that $(b \cdot m/n)^{d'} \leq 1$, for every $d' \geq 0$, with equality holding only for $d' = 0$.

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 corresponds to the bilinear form

$$\sum_{i\in T_1} \sum_{(k_1,...,k_{t'})\in S} F_i(\mathtt{u}(I_1(k_1)),...,\mathtt{u}(I_{t'}(k_{t'})),x^{(t'+2)}) \cdot G_i(\mathtt{u}(I_1(k_1)),...,\mathtt{u}(I_{t'}(k_{t'})),x^{(t'+1)}) \qquad (11)$$

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

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 $\mathtt{BL}(i) \cap [t']$. Then, Eq. (10) is re-written as

$$\sum_{k_{\mathtt{BL}_1(i)},...,k_{\mathtt{BL}_{d_i}(i)}\in[b]} G'_i(\mathtt{u}(I_{\mathtt{BL}_1(i)}(k_{\mathtt{BL}_1(i)})),...,\mathtt{u}(I_{\mathtt{BL}_{d_i}(i)}(k_{\mathtt{BL}_{d_i}(i)})),x^{(t'+1)}) \cdot F_{i,S(\overline{I})}^{(k_{\mathtt{BL}_1(i)},...,k_{\mathtt{BL}_{d_i}(i)})}(x^{(t'+2)}) \quad (12)$$

where G'_i and $F_{i,S}^{(k_1,...,k_{d_i})}$ are defined in an analogous matter.[8] 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-subserts,

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

[8] 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

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

the expected rank of the matrix that corresponds to the bilinear form of Eq. (11) 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. (12).)[9]

Gates of Type 2: $t'+2 \in \mathtt{BL}(i)$ and $|\mathtt{BL}(i)\cap[t']| \geq d$. We consider the same $b^{t'}$ random assignments as before. Recall that 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 $\mathtt{u}(i) = 0^{i-1}10^{n-i}$. Recall that, for each $j \in \mathtt{BL}(i)\cap[t']$, the probability that I_j hits any of the variables of block j that feeds G_i is at most $b\cdot m/n$, and so 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 $d \geq 2/\epsilon$).

(As in the analysis of Type 1, each gate G_i of Type 2 vanishes under all $b^{t'}$ random assignment with probability at least $1-(b\cdot m/n)^{|\mathtt{BL}(i)\cap[t']|}$. Here, being guaranteed that $|\mathtt{BL}(i)\cap[t']| \geq d$, for a sufficiently large constant d, allows us to totally ignore all these gates.)

(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 form 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 corresponding n-by-n matrix may be arbitrary, unless there are cancellations between the $b^{t'}$ random restrictions. Indeed, the entire point of choosing many random restrictions was to form such cancellations.)

Gates of Type 3: $t'+2 \in \mathtt{BL}(i)$ and $|\mathtt{BL}(i)\cap[t']| < d$. 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 that $b^{t'} > m$, 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

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

[9] Specifically, using

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

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 $(m/n)^{d_i}$.

where we rely on the fact that F_i is not fed by variables of block $t'+2$ (since $t'+2 \in \mathtt{BL}(i)$ by definition of Type 3). This means that these G_i's do not contribute to the corresponding bilinear form, since the 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., $T'_3 = \{i : \mathtt{BL}(i) = \{t'+1, t'+2\}\}$), the following expression is identially zero.

$$\sum_{i \in T'_3} \sum_{(k_1,...,k_{t'}) \in S} F_i(\mathtt{u}(I_1(k_1)), ..., \mathtt{u}(I_{t'}(k_{t'}))) \cdot G_i(\mathtt{u}(I_1(k_1)), ..., \mathtt{u}(I_{t'}(k_{t'})), x^{(t'+1)}, x^{(t'+1)}) \qquad (13)$$

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 Eq. (13), 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.

Specifically, for any fix choice of the random sequence of b-subsets, $\overline{I} = (I_1, ..., I_{t'})$, we consider an auxiliary $b^{d-1} \cdot |T_3|$-by-$b^{t'}$ Boolean matrix in which the rows correspond to pairs $(i, (k'_1, ..., k'_{d-1})) \in T_3 \times [b]^{d-1}$, the columns correspond to choices of $\overline{k} = (k_1, ..., k_{t'}) \in [b]^{t'}$, and the value of entry $((j, (k'_1, ..., k'_{d-1})), \overline{k})$ is determined according to $F'_i(\mathtt{u}(k_1^{(\sigma_1)}), ..., \mathtt{u}(k_{t'}^{(\sigma_{t'})}))$, provided that $(k'_1, ..., k'_{d-1})$ fits $\overline{k}$ (i.e., $k'_j = k_{\mathtt{BL}_j(i)}$ for every $j \in [d_i]$). Specifically:

- For each $i \in T_3$, let $d_i \stackrel{\text{def}}{=} |\mathtt{BL}(i) \cap [t']| < d$ and $\mathtt{BL}_j(i)$ be the j^{th} element in $\mathtt{BL}(i) \cap [t']$. We actually consider only b^{d_i} rows that correspond to i (and let the other $b^{d-1} - b^{d_i}$ rows be set to $0^{b^{t'}}$). These rows will correspond to all choices 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 respresents a choice of a sequence in $I_1 \times \cdots \times I_{t'}$ (i.e., the choice that determines the sequence $(I_1(k_1), ..., I_{t'}(k_{t'})) \in [n]^{t'}$, which in turn determines the assignment $(\mathtt{u}(I_1(k_1)), ..., \mathtt{u}(I_{t'}(k_{t'})))$.

  (Indeed, while the foregoing $(k'_1, ..., k'_{d_i})$ represents the choices made for the blocks appearing in $\mathtt{BL}(i) \cap [t']$, the choices $\overline{k} = (k_1, ..., k_{t'})$ represent the choices made for all blocks in $[t']$. Our focus will be on $(k'_1, ..., k'_{d_i})$'s that fit the various $\overline{k}$'s.)

- The value of entry $((i, (k'_1, ..., k'_{d_i})), \overline{k})$ 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
  - $k'_j = k_{\mathtt{BL}_j(i)}$ for every $j \in [d_i]$ (i.e., $(k'_1, ..., k'_{d_i})$ fits $\overline{k}$).

  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 $((i, (k'_1, ..., k'_{d_i})), \overline{k})$ is 0 in case $(k'_1, ..., k'_{d-1})$ does not fits $\overline{k}$ (i.e., $k'_j \neq k_{\mathtt{BL}_j(i)}$ 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 by $(k'_1, ..., k'_{d-1})$ matter.

Suppose, for simplicity, that the foregoing auxiliary matrix contains an all-zero column that is indexed $\overline{k}$. This means that for every $i \in T_3$ and the unique $(k'_1, ..., k'_{d_i})$ that fits $\overline{k}$ (i.e., $k'_j =$

$k_{\mathtt{BL}_j(i)}$ for every $j \in [d_i]$), it holds that $F_i(\mathtt{u}(I_1(k_1)),...,\mathtt{u}(I_{t'}(k_{t'}))) = 0$, because the corresponding $((i,(k'_1,...,k'_{d_i})),\overline{k})$-entry of the matrix is 0 (whereas $(k'_1,...,k'_{d_i})$ does fit $\overline{k}$). Using the notation

$$F_i^{(\overline{k})}(y,z) \stackrel{\text{def}}{=} F_i(\mathtt{u}(I_1(k_1)),...,\mathtt{u}(I_{t'}(k_{t'}))) \cdot G_i(\mathtt{u}(I_1(k_1)),...,\mathtt{u}(I_{t'}(k_{t'})),y,z) \tag{14}$$

it follows that $\sum_{i\in T_3} F_i^{(\overline{k})}(y,z)$ is identically zero.

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

$$\sum_{\overline{k}\in S:(\forall j\in[d_i])\, k_{\mathtt{BL}_j(i)}=k'_j} F_i^{(\overline{k})}(y,z) \tag{15}$$

$$= \; G'_i(\mathtt{u}(I_1(k'_1)),...,\mathtt{u}(I_{d_i}(k_{d_i})),y,z) \cdot \sum_{\overline{k}\in S:(\forall j\in[d_i])\, k_{\mathtt{BL}_j(i)}=k'_j} F_i(\mathtt{u}(I_1(k_1)),...,\mathtt{u}(I_{t'}(k_{t'}))) \tag{16}$$

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 is fit by $(k'_1,...,k'_{d_i})$ (i.e., $k'_j = k_{\mathtt{BL}_j(i)}$ for every $j \in [d_i]$).

The punch-line is that Eq. (16) 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 $(k'_1,...,k'_{d_i})$, 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 $(k'_1,...,k'_{d_i})$). 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 $(k'_1,...,k'_{d_i})$, we have

$$\begin{aligned}\sum_{\overline{k}\in S:(\forall j\in[d_i])\, k_{\mathtt{BL}_j(i)}=k'_j} F_i(\mathtt{u}(I_1(k_1)),...,\mathtt{u}(I_{t'}(k_{t'}))) &= \sum_{\overline{k}\in S} M_{(i,(k'_1,...,k'_{d_i}),\overline{k}} \\ &= 0.\end{aligned}$$

Hence, Type 3 gates have no contribution to the bilinear function $B_S : GF(2)^{n+n} \to \mathrm{GF}(2)$ defined as

$$B_S(y,z) \stackrel{\text{def}}{=} \sum_{\overline{k}=(k_1,...,k_{t'})\in S} F(\mathtt{u}(I_1(k_1)),...,\mathtt{u}(I_{t'}(k_{t'})),y,z) \tag{17}$$

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, and $S = S(\overline{I})$ is chosen based on $\overline{I}$.

Recall that the contribution of Type 2 gates vanishes with very high probability, and the contribution of Type 1 gates corresponds to 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 $B_{S(\overline{I})}$ corresponds to a matrix of rank at most $5m$.

2.4 Wrapping-up and reaching a contradiction

We have essentially established the following Lemma 2.1, 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 2.1 (low AN2-complexity of F implies low rank of 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 a t'-long sequence of b-subsets, $\overline{I} = (I_1, ..., I_{t'})$, there exists a non-empty set $S \subseteq [b]^{t'}$ such that the matrix corresponding to the bilinear function B_S of* Eq. (17) *has rank at most $5m$. Formally, we refer to the bilinear form*

$$B_S^{(\overline{s},\overline{I})}(y,z) \stackrel{\text{def}}{=} \sum_{\overline{k}=(k_1,...,k_{t'})\in S} F(\mathtt{u}(I_1(k_1)), ..., \mathtt{u}(I_{t'}(k_{t'})), y, z, s^{(t'+3)}, ..., s^{(t'+t'')}), \tag{18}$$

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 in 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. (8). Recall that we have fixed $\overline{s} = (s^{(t'+3)}, ..., s^{(t'+t'')}) \in \mathrm{GF}(2)^{(t''-2)\cdot n}$ upfront, and all we did referred to that fixed $\overline{s}$ (and holds for any such $\overline{s}$). We have broken the sum in Eq. (8) into three parts, corresponding to the three types of gates, and analyzed each type separately.

For Type 1, we showed 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 $B_S^{(\overline{s},\overline{I})}$ corresponds to a matrix of expected rank at most m. (This used $b \leq (n/m)$, which holds under the hypothesis.) Next, we showed that, over the same random choice of $\overline{I}$, and for every $S \subseteq [b]^{t'}$, the contribution of gates of Type 2 to $B_S^{(\overline{s},\overline{I})}$ corresponds to 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 $0.8 - 0.1$, the contribution of gates of Types 1 and 2 to $B_S^{(\overline{s},\overline{I})}$ corresponds to a matrix of rank at most $5m$.

Lastly, considering the Type 3 gates, we showed 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 $B_S^{(\overline{s},\overline{I})}$ corresponds to an all-zero matrix. Here we used $t' \geq d + \log_b m$. Hence, using $b \leq (n/m)^{1/2}$ (and $d = 2\log_{n/m} n$), it suffices to have $t' \geq 2\log_{n/m} n + \log_b m$, which holds by the hypothesis.

Formally, for $\overline{s} = (s^{(t'+3)}, ..., s^{(t'+t'')})$, and $\overline{I} = (I_1, ..., I_{t'}) \in \binom{[n]}{b}^{t'}$, letting $\overline{\mathtt{u}}(\overline{k}) = (\mathtt{u}(I_1(k_1)), ..., \mathtt{u}(I_{t'}(k_{t'})))$, we re-write Eq. (18) as

$$B_S^{(\overline{s},\overline{I})}(y,z) = \sum_{\overline{k}=(k_1,...,k_{t'})\in S} F(\overline{\mathtt{u}}(\overline{k}), y, z, s^{(t'+3)}, ..., s^{(t'+t'')}). \tag{19}$$

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

$$\begin{aligned} B_S^{(\overline{s},\overline{I})}(y,z) &= \sum_{\overline{k}\in S} \sum_{i\in[m]} F_i(\overline{\mathtt{u}}(\overline{k}), y, z, \overline{s}) \cdot G_i(\overline{\mathtt{u}}(\overline{k}), y, z, \overline{s}) \\ &= \sum_{\overline{k}\in S} \sum_{i\in T_1} F_i(\overline{\mathtt{u}}(\overline{k}), y, z, \overline{s}) \cdot G_i(\overline{\mathtt{u}}(\overline{k}), y, z, \overline{s}) \qquad (20) \\ &\quad + \sum_{\overline{k}\in S} \sum_{i\in T_2} F_i(\overline{\mathtt{u}}(\overline{k}), y, z, \overline{s}) \cdot G_i(\overline{\mathtt{u}}(\overline{k}), y, z, \overline{s}), \qquad (21) \end{aligned}$$

since the sum that corresponds to T_3 is identically zero due to the choice of S. Recalling that, with probability at least $4/5$, the matrix corresponding to Eq. (20) has rank at most $5m$, and that with probability at least $9/10$ the matrix corresponding to Eq. (21) is identically zero, the lemma follows. ∎

Reaching a contradiction. Lemma 2.1 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 corresponding to 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 2.2 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 corresponding to 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 $\mathtt{AN}_2(F) = \Omega(n^{1-\epsilon})$, and Theorem 1.7 follows. But let us first prove the following lemma.

Lemma 2.2 (typically 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 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 corresponding to the bilinear function $B_S^{(\overline{s},\overline{I})}$ of* Eq. (19) *has rank $\Omega(n)$.*

Proof: For every $\overline{I} \in \binom{[n]}{b}^{t'}$ 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 $\overline{\mathtt{u}}(\overline{k}) = (\mathtt{u}(I_1(k_1)1), ..., \mathtt{u}(I_{t'}(k_{t'})))$, observe that

$$\begin{aligned}
B_S^{(\overline{s},\overline{I})}(y,z) &= \sum_{\overline{k}\in S} F(\overline{\mathtt{u}}(\overline{k}), y, z, \overline{s}) \\
&= \sum_{\overline{k}\in S} \sum_{(i_1,...,i_{t'+2})\in \overline{I}\times[n]^2} G_{\mathtt{sb}}(\overline{s})_{(i_1,...,i_{t'+2})} \cdot \prod_{j\in[t']} (\mathtt{u}(I_j(k_j)))_{i_j} \cdot y_{i_{t'+1}} \cdot z_{i_{t'+2}} \qquad (22) \\
&= \sum_{\overline{k}\in S} \sum_{(i_{t'+1},i_{t'+2})\in[n]} G_{\mathtt{sb}}(\overline{s})_{(I_1(k_1),...,I_{t'}(k_{t'}),i_{t'+1},i_{t'+2})} \cdot y_{i_{t'+1}} \cdot z_{i_{t'+2}} \qquad (23)
\end{aligned}$$

where the last equality holds because only t'-tuples $(i_1, ..., i_{t'})$ that satisfy $i_j = I_j(k_j)$ contribute to Eq. (22). The difference between Eq. (22) and Eq. (23) 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 corresponds to $B_S^{(\overline{s},\overline{I})}$ 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 $2^{-n}+2^{-n}$.[10] 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 2.2 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 $S \subseteq [b]^{t'}$, the matrix corresponding to $B_S^{(\overline{s},\overline{I})}$ has rank $\Omega(n)$. On the other hand, Lemma 2.1 implies that, under the same probability space, with probability at least $2/3$, there exists a set S such that the matrix corresponding to $B_S^{(\overline{s},\overline{I})}$ has rank at most $5m$. 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 $\beta \le 1/2$, and setting $t' = 2\log_{n/\mathtt{AN}_2(F)} n + \log_b \mathtt{AN}_2(F)$, 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 $t' = 2\log_{n^\epsilon} n + \log_{(n^\epsilon)^\beta} n = O(1/\epsilon^2)$. Theorem 1.7 follows by using an adequate small-bias generator, as provided by Theorem A.6.

Remark 2.3 (using rigidity rather than pure rank): *We note that the proof of Lemma 2.1 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 2.2 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 $b\ge 2$. 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 Conclusions

Theorem 1.7 can be extended to $\epsilon=\epsilon(n)$ that vanishes with n, provided that $\epsilon(n)\ge\sqrt{2/\log_2 n}$, since the argument presupposes that $b=n^{\epsilon\cdot\beta}=n^{\epsilon^2/2}$ is at least 2. Hence, we actually have

Theorem 3.1 (Theorem 1.7, rephrased): *For every $\epsilon:\mathbb{N}\to(0,1)$ such that $\epsilon(n)\ge\sqrt{2/\log_2 n}$, 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 $\mathtt{AN}_2(f)=\Omega(n^{1-\epsilon(n)})$.*

[10] 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]).

Theorem 3.1 does not allow setting $\epsilon(n) = \Omega(1/\log n)$ and deriving an $\Omega(n)$ lower bound, but such a lower bound would have missed the target of being truly linear in the length of the input (i.e., $\mathrm{poly}(1/\epsilon(n)) \cdot n$) and being truly explicit (i.e., the function is $n^{\mathrm{poly}(1/\epsilon(n))}$-time computable). 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.

   For starters, show that functions as in Theorem 1.7 can be computed in fixed polynomial-time, rather than in $O(n^{\mathrm{poly}(1/\epsilon)})$-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 pressing challenge is to address the first part of Problem 1.6; 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 $\mathtt{AN}(f) = \Omega(n^{0.99})$.

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. Lastly, I thank Avi Wigderson for useful comments regarding this work.

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. *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] 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.

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

[10] 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 ingrediants 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.

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 superlinear 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): *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*

$$\frac{1}{2} \cdot \left| \mathrm{E}\left[(-1)^{T(G(U_k))}\right] - \mathrm{E}\left[(-1)^{T(U_{s(k)})}\right] \right| \leq \epsilon(k). \tag{24}$$

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. (24) equals the total variation distance between the "verdict" of T in the two cases (i.e., the statiustical 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. The general case 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

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

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 [10], 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 one. 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 [10]); 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.

Here we can use 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). 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 has bias at most $D_2 \cdot \epsilon_1(k/D_2)$ with respect to tests of degree d_1. We just use 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 $U_{s_1(k/D_2)}$. 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 $H_0 = U_{D_2 \cdot s_1(k/D_2)}$ and $H_{D_2} = G'_1(U_k)$, whereas 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 straighhtforward. 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. ■