Learnability Beyond AC^0

Jeffrey C. Jackson*
Department of Mathematics and Computer Science
Duquesne University
Pittsburgh, PA 15282
jackson@mathcs.duq.edu

Adam R. Klivans†
Department of Mathematics
MIT
Cambridge, MA 02139
klivans@math.mit.edu

Rocco A. Servedio‡
Division of Engineering and Applied Sciences
Harvard University
Cambridge, MA 02138
rocco@deas.harvard.edu

Abstract

We give an algorithm to learn constant-depth polynomial-size circuits augmented with majority gates under the uniform distribution using random examples only. For circuits which contain a polylogarithmic number of majority gates the algorithm runs in quasipolynomial time. This is the first algorithm for learning a more expressive circuit class than the class AC^0 of constant-depth polynomial-size circuits, a class which was shown to be learnable in quasipolynomial time by Linial, Mansour and Nisan in 1989. Our approach combines an extension of some of the Fourier analysis from Linial *et al.* with hypothesis boosting. We also show that under a standard cryptographic assumption our algorithm is essentially optimal with respect to both running time and expressiveness (number of majority gates) of the circuits being learned.

*This material is based upon work supported by the National Science Foundation under Grant No. CCR-9877079.
†Supported in part by NSF grant CCR-97-01304.
‡Supported by NSF Grant CCR-98-77049 and by an NSF Mathematical Sciences Postdoctoral Research Fellowship.

1 Introduction

1.1 Motivation

In his seminal 1984 paper "A Theory of the Learnable" Valiant proposed the Probably Approximately Correct (PAC) model of learning in an attempt to understand what programs a machine can acquire without explicit instruction. A central question that arises from this paradigm is the following: what is the most expressive class of Boolean functions that can be learned by a computationally efficient algorithm? As stated by Valiant, "The results of *learnability theory* ... indicate the maximum granularity of the single concepts that can be acquired without programming" [28]. Working towards this goal, researchers in computational learning theory have developed many algorithms for learning various classes of Boolean circuits. This paper gives a new algorithm to learn the most expressive class of Boolean circuits considered to date for a well-studied variant of Valiant's model.

1.2 Previous Results

In Valiant's original model, a class of functions is said to be efficiently learnable if there is a polynomial-time algorithm which succeeds given labeled examples drawn from an arbitrary probability distribution. Unfortunately, few classes are known to be efficiently learnable in this sense. Hence researchers have considered algorithms which run in superpolynomial time and/or succeed under specific distributions on examples such as the uniform distribution.

Several subexponential time algorithms are known for learning interesting classes in the distribution-free model where the learning algorithm must succeed under an arbitrary probability distribution. Ehrenfeucht and Haussler [13] and Blum [5] have given $n^{\log s}$ time algorithms for learning size-s decision trees over n Boolean variables. Improving on results of Bshouty [9] and Tarui and Tsukiji [27], Klivans and Servedio [21] have recently given an algorithm for learning s-term Disjunctive Normal Form (DNF) formulae in time $2^{n^{1/3} \log s \log n}$.

Stronger results can be achieved by requiring the learning algorithm to succeed only when given examples drawn from the uniform distribution on the Boolean cube. One such uniform distribution learning algorithm is Jackson's Harmonic Sieve algorithm for learning s-term DNF formulae [17]. The Sieve runs in $\text{poly}(n, s)$ time but requires that the learning algorithm be allowed to make *membership queries* for the value of the target function on specified points. Another well-known uniform distribution learning algorithm is the algorithm of Linial *et al.* [23] which learns size-s depth-d AND/OR/NOT circuits in time $n^{(\log s)^d}$ without using membership queries. This result is noteworthy for several reasons: (1) The class AC^0 of constant-depth polynomial-size circuits is highly expressive relative to other circuit classes studied in learning theory – note that DNF formulae are AC^0 circuits of depth 2. (2) Kharitonov [19] has given compelling evidence that the Linial *et al.* algorithm is essentially optimal by showing that an $n^{(\log s)^{o(d)}}$-time algorithm for learning size-s depth-d circuits under uniform would contradict a plausible cryptographic assumption about the hardness of integer factorization.

1.3 Our Results

We give the first algorithm to learn constant-depth circuits augmented with majority gates. Although Kharitonov proved that the running time of Linial *et al.*'s algorithm is essentially optimal for constant-depth AND/OR/NOT circuits, our results show that it is possible to learn a strictly more expressive class of circuits in the same amount of time.[1] We also show that under Kharitonov's cryptographic assumption our algorithm is essentially optimal with respect to both running time and expressiveness (number of majority gates) of the functions being learned.

We believe that augmenting constant depth circuits with a limited number of majority gates is an interesting and natural extension of AC^0. Such circuits have been studied by several researchers in complexity theory [1, 3, 4] and represent a significant extension of AC^0 (recall that a single majority of n variables requires depth-d circuits of size $2^{n^{1/d}}$ [16]). Our main positive result for these circuits can be stated informally as follows (a precise statement is given in Section 3.3):

Theorem 1 *Quasipolynomial ($2^{polylog\, n}$) size constant-depth circuits which contain a polylogarithmic number of majority gates can be learned under the uniform distribution in quasipolynomial time from random examples only.*

As we allow constant depth circuits to contain more and more majority gates we move toward the circuit complexity class TC^0 which can be viewed as AC^0 augmented with a polynomial number of majority gates. TC^0 is a highly expressive class which contains functions such as integer multiplication and division – indeed it is currently conceivable that all of $\#\mathsf{P}$ is contained in non-uniform TC^0. Naor and Reingold [25] have constructed pseudo-random functions in TC^0 (under a widely held cryptographic assumption) and thus it seems unlikely that effective learning algorithms can be given for TC^0. Using results from [25] we establish the following lower bound for learning which holds under Kharitonov's plausible cryptographic assumption (a precise statement is given in Section 4):

Theorem 2 *Any algorithm that even weakly learns depth-5 circuits containing more than a polylogarithmic number of majority gates under the uniform distribution must run in more than quasipolynomial time, even if membership queries are allowed.*

Theorems 1 and 2 together give a fairly complete picture of the complexity of learning constant depth circuits which contain majority gates.

1.4 Our Approach

The learning algorithm of Linial *et al.* for AC^0 is based on the fact that any function computed by a constant-depth polynomial-size circuit can be well approximated by the low-order terms in its Fourier representation. More precisely, the main lemma of [23] shows that any AC^0 function on n variables can have at most an ϵ fraction of its Fourier spectrum weight on sets which contain more than $\mathrm{polylog}(n/\epsilon)$ variables. This property need not hold for circuits which contain majority gates; Bshouty and Tamon [11] have shown that the majority function itself has an $\Omega(1/\sqrt{n})$ fraction of its Fourier spectrum weight on sets of size $\Omega(\sqrt{n})$. Hence a different approach seems to be required for circuits which contain majority gates.

Our algorithm learns using a stagewise approach based on hypothesis boosting. Boosting is a well known technique in computational learning theory which can be used to transform a weak learning algorithm (which constructs an approximator to the function being learned whose error is only slightly less than 1/2) into a strong learning algorithm which generates a highly accurate hypothesis. Boosting algorithms accomplish this by running the weak learning algorithm repeatedly on a sequence of carefully constructed distributions.

[1] We show in Section 3.4 that our algorithm runs slightly faster asymptotically than the Linial *et al.* algorithm for learning standard constant-depth AND/OR/NOT circuits.

We show (Lemma 6) that for any function f in our class and any distribution $\mathcal{D}$ which is not too far from uniform, there exists a low-order parity function which is weakly correlated with f. Since this parity depends on only a small number of variables it can be found using exhaustive search. As we show in Section 3, the size bound for this parity – and hence the running time of our algorithm – depends directly on the extent of $\mathcal{D}$'s deviation from the uniform distribution. Fortunately, boosting algorithms are known which have the property that if the initial distribution over examples is uniform then the distributions constructed by the boosting algorithm will not deviate too far from uniform. Using such a boosting algorithm we transform our weak learning algorithm into a strong learning algorithm which generates a highly accurate approximator to the unknown target function.

The form of the hypothesis output by our algorithm is a majority over quasipolynomially many parity functions. Since the Harmonic Sieve is already capable of learning this class in quasipolynomial time [17], it is natural to ask whether our new algorithm is learning a class that could not be learned by the existing Sieve. Several points should be noted. First, unlike the Sieve, our algorithm does not require membership queries, which is a significant advantage. Second, while our results imply that the Sieve is capable of learning AC^0 circuits augmented with a polylogarithmic number of majority gates, this was not known prior to the analysis contained in this paper. In particular, the most general prior results [17] showed that the Sieve was capable of learning a majority of s parity functions in time polynomial in s. However, since even the class AC^0 itself is known to contain a linear-size circuit C such that any majority of parities representing C exactly contains exponentially many distinct parity functions [22], these prior results do not imply our results.

1.5 Learning Other Extensions of AC^0

Another natural question raised by our work is whether it is possible to efficiently learn constant-depth circuits which contain a limited number of gates other than majority. In Section 5 we consider learning constant-depth $AND/OR/NOT$ circuits augmented with a single additional gate at the root which may be either a parity gate, a MOD_p gate for prime $p > 2$, or a threshold gate which computes an arbitrary weighted majority of its inputs. We give evidence that with respect to known techniques each of these three simple extensions of AC^0 may be difficult to learn.

1.6 Organization

In Section 3 we give our algorithm for learning constant-depth circuits with majority gates. In Section 4 we give cryptographic hardness results for learning constant-depth circuits with majority gates. Section 5 discusses prospects for learning other extensions of AC^0.

2 Preliminaries

We write $[n]$ to denote the set $\{1, \ldots, n\}$, $\mathcal{U}$ to denote the uniform distribution over $\{-1,1\}^n$ and $\log x$ to denote $\log_2 x$. We view Boolean functions as mappings from $\{-1,1\}^n$ to $\{-1,1\}$ where -1 represents TRUE and 1 represents FALSE. Unless otherwise indicated probabilities and expectations are with respect to $\mathcal{U}$.

Recall that AC^0 is the class of constant-depth polynomial-size circuits composed of unbounded fan-in AND, OR and NOT gates. The majority function $MAJ : \{-1,1\}^n \to \{-1,1\}$ is defined by $MAJ(x) = \mathrm{sign}(x_1 + \cdots + x_n)$ which is 1 if and only if $x_1 + \cdots + x_n \geqslant 0$. We write MAC^0 to denote the class of constant-depth polynomial-size circuits which have a single unbounded fan-in majority gate at the root, and we refer to such circuits as *majority of* AC^0.

For $A \subseteq [n]$ the parity function $\chi_A : \{-1,1\}^n \to \{-1,1\}$ is defined by $\chi_A(x) = \prod_{i \in A} x_i$. As is well known, with inner product $\langle f, g\rangle = E[fg]$ and norm $\|f\| = \sqrt{E[f^2]}$ the 2^n parity functions $\{\chi_A\}_{A \subseteq [n]}$ form an orthonormal basis for the vector space of real-valued functions on $\{-1,1\}^n$. Consequently every real-valued function $f : \{-1,1\}^n \to \{-1,1\}$ can be uniquely expressed as a linear combination $f(x) = \sum_{A \subseteq [n]} \hat{f}(A)\chi_A(x)$. The coefficients $\hat{f}(A)$ are known as the *Fourier spectrum* of f. Orthonormality implies that $\hat{f}(A) = \langle f, \chi_A\rangle$ and thus $\hat{f}(A)$ measures the correlation between f and χ_A. Orthonormality also implies *Parseval's identity*, which states that for every real-valued function f we have $E[f^2] = \sum_{A \subseteq [n]} \hat{f}(A)^2$. For Boolean functions we thus have $\sum_{A \subseteq [n]} \hat{f}(A)^2 = 1$. The following easily verified fact is from [23]:

Fact 3 *For any Boolean function f and real-valued function g,*

$$\Pr[f(x) \neq sign(g(x))] \leqslant E[(f(x) - g(x))^2] = \sum_{A \subseteq [n]} (\hat{f}(A) - \hat{g}(A))^2.$$

The equality follows from Parseval's identity and the linearity of the Fourier transform.

Finally we note the following simple but useful fact: for any distribution $\mathcal{D}$, if f, h are ± 1-valued functions then $\Pr_{\mathcal{D}}[f = h] = 1/2 + E_{\mathcal{D}}[fh]/2$.

2.1 The Learning Model

The learning model we consider is a well-studied distribution-specific variant of Valiant's Probably Approximately Correct (PAC) learning model [28]. Let f be a Boolean function and $\mathcal{D}$ a probability distribution on $\{-1,1\}^n$. We write $EX(f, \mathcal{D})$ to denote a random example oracle which when invoked outputs a labeled example $\langle x, f(x)\rangle$ where $x \in \{-1,1\}^n$ is chosen according to $\mathcal{D}$.

Let C be a class of Boolean functions on $\{-1,1\}^n$. Let $f \in C$ be an unknown member of C and let A be a learning algorithm which takes as input accuracy and confidence parameters ϵ, δ and can invoke the oracle $EX(f, \mathcal{D})$. We say that *A learns C under the uniform distribution* if A satisfies the following condition: for all $0 < \epsilon, \delta < 1$, if A is given access to $EX(f, \mathcal{U})$ then with probability at least $1 - \delta$ A outputs a hypothesis $h : \{-1,1\}^n \to \{-1,1\}$ such that $\Pr[f(x) = h(x)] \geqslant 1 - \epsilon$.

In order to obtain our uniform distribution learning algorithm we will need to consider *weak* learning algorithms which work for a broader range of distributions but can only generate hypotheses which are slightly more accurate than random guessing. For $\gamma > 0$ we say that A is a *weak learning algorithm for C with advantage γ* if A satisfies the following condition: for any distribution $\mathcal{D}$, for any $0 < \delta < 1$, if A is given access to $EX(f, \mathcal{D})$ then with probability at least $1 - \delta$ A outputs a hypothesis $h : \{-1,1\}^n \to \{-1,1\}$ such that $\Pr_{\mathcal{D}}[f(x) = h(x)] \geqslant 1/2 + \gamma$. As we will see the advantage γ can depend on parameters such as n, the size of a circuit for f, and the L_∞ norm of the distribution $\mathcal{D}$.

A *boosting algorithm* `B` is an algorithm which satisfies the following condition: for any Boolean function f and any distribution $\mathcal{D}$ over $\{-1,1\}^n$, if `B` is given $0 < \epsilon, \delta < 1$, $0 < \gamma < 1/2$, an example oracle EX$(f, \mathcal{D})$ and a weak learning algorithm `WL` with advantage γ, then with probability at least $1 - \delta$ algorithm `B` outputs a hypothesis h such that $\Pr_{\mathcal{D}}[f(x) = h(x)] \geqslant 1 - \epsilon$. We say that a *canonical boosting algorithm* `B` is a boosting algorithm which has the following iterative structure:

- At stage 0 `B` starts with $\mathcal{D}_0 = \mathcal{D}$ and uses `WL` to generate with high probability a hypothesis h_0 satisfying $\Pr_{\mathcal{D}_0}[f(x) = h(x)] \geqslant 1/2 + \gamma$.
- At stage i `B` does two things: (1) constructs a new distribution $\mathcal{D}_i$ which favors points where the previous hypotheses $h_0, \ldots, h_{i-1}$ do poorly at predicting the value of f, and (2) runs `WL`

using the simuated example oracle $\mathrm{EX}(f, \mathcal{D}_i)$ to produce with high probability a hypothesis h_i satisfying $\Pr_{\mathcal{D}_i}[f(x) = h_i)] \geqslant 1/2 + \gamma$.

- Finally, after doing this repeatedly for some number T of stages, B combines the hypotheses $h_0, \ldots, h_{T-1}$ in some way to obtain with high probability a strong final hypothesis h satisfying $\Pr_{\mathcal{D}}[f(x) = h(x)] \geqslant 1 - \epsilon$.

3 The algorithm

We first present a weak learning algorithm for the class of constant-depth circuits with a single majority gate at the root (Section 3.1). In Section 3.2 we show how to boost this weak learning algorithm into a strong uniform distribution learning algorithm for MAC^0. In Section 3.3 we observe that our algorithm can in fact be used to learn constant-depth circuits which contain several majority gates. Finally in Section 3.4 we show that the new algorithm for MAC^0 can be used to improve previous results of Linial *et al.* for learning and approximating AC^0.

3.1 Weak learning MAC^0 under smooth distributions

Let $f : \{-1,1\}^n \to \{-1,1\}$ be computed by an MAC^0 circuit of size s and depth $d+1$, so $f = MAJ(C_1, \ldots, C_t)$ where $t \leqslant s$ and each $C_i : \{-1,1\}^n \to \{-1,1\}$ is a circuit of depth at most d and size at most s. Without loss of generality we can assume that $C_1(x) + \ldots + C_t(x) \neq 0$ for all $x \in \{-1,1\}^n$. (If this sum is ever zero we can reexpress f as $MAJ(C_1, \ldots, C_t, C_1, \ldots, C_t, 1)$. This leaves the depth unchanged and at most doubles the size which makes no difference for our asymptotic results.) We recall the "discriminator lemma" of Hajnal *et al.* [15]:

Lemma 4 *Let $\mathcal{H}$ be a class of ± 1-valued functions and let $f : \{-1,1\}^n \to \{-1,1\}$ be expressible as $MAJ(h_1, \ldots, h_t)$ where each $h_i \in \mathcal{H}$ and $h_1(x) + \cdots + h_t(x) \neq 0$ for all x. Then for any distribution $\mathcal{D}$ over $\{-1,1\}^n$ there is some $h \in \mathcal{H}$ such that $|E_{\mathcal{D}}[fh]| \geqslant 1/t$.*

Hence for any distribution $\mathcal{D}$ there is a size-s depth-d circuit $C_{\mathcal{D}}$ such that $E_{\mathcal{D}}[fC_{\mathcal{D}}] \geqslant 1/s$.

Our goal is to show that as long as $\mathcal{D}$ does not assign too much probability weight to any single point then there is some parity function χ_A (where A is small as described below) which has nonnegligible correlation with f under $\mathcal{D}$. Toward this end we use the main lemma of Linial *et al.* [23]:

Lemma 5 *Let C be a circuit of size s and depth d. Then for any integer k,*

$$\sum_{|A|>k} \hat{C}(A)^2 \leqslant 2s2^{-k^{1/d}/20}.$$

Fix a distribution $\mathcal{D}$ on $\{-1,1\}^n$. Let $k = (20 \log(8s^3 2^n L_\infty(\mathcal{D})))^d$ and let $g : \{-1,1\}^n \to \mathbb{R}$ be defined by $g(x) = \sum_{|A| \leqslant k} \hat{C}_{\mathcal{D}}(A)\chi_A(x)$. Then by Parseval's identity and Lemma 5,

$$E[(g(x) - C_{\mathcal{D}}(x))^2] = \sum_{A \subseteq [n]} (\hat{g}(A) - \hat{C}_{\mathcal{D}}(A))^2 = \sum_{|A|>k} \hat{C}_{\mathcal{D}}(A)^2 \leqslant \frac{1}{4s^2 2^n L_\infty(\mathcal{D})}.$$

Since $\mathcal{D}(x) \leqslant L_\infty(\mathcal{D})$ for all x, we have $E_{\mathcal{D}}[(g(x) - C_{\mathcal{D}}(x))^2] \leqslant 1/4s^2$. Since $E_{\mathcal{D}}[X^2] \geqslant E_{\mathcal{D}}[X]^2$ for any distribution $\mathcal{D}$ and any random variable X, we have $E_{\mathcal{D}}[|g(x) - C_{\mathcal{D}}(x)|] \leqslant 1/2s$. Thus

$$\begin{aligned} E_{\mathcal{D}}[fg] &= E_{\mathcal{D}}[fC_{\mathcal{D}}] + E_{\mathcal{D}}[f(g - C_{\mathcal{D}})] \\ &\geqslant E_{\mathcal{D}}[fC_{\mathcal{D}}] - E_{\mathcal{D}}[|g - C_{\mathcal{D}}|] \\ &\geqslant 1/s - 1/2s = 1/2s \qquad (1) \end{aligned}$$

where the first inequality holds because the range of f is $\{-1,1\}$. Thus the real-valued function g has nonnegligible correlation with the MAC^0 function f under $\mathcal{D}$.

Using linearity of expectation, we have

$$\sum_{|A|\leqslant k} |\hat{C}_{\mathcal{D}}(A)||E_{\mathcal{D}}[f\chi_A]| \geqslant \sum_{|A|\leqslant k} \hat{C}_{\mathcal{D}}(A)E_{\mathcal{D}}[f\chi_A] = E_{\mathcal{D}}[f \sum_{|A|\leqslant k} \hat{C}_{\mathcal{D}}(A)\chi_A] = E_{\mathcal{D}}[fg] \geqslant 1/(2s).$$

Since each Fourier coefficient $\hat{C}_{\mathcal{D}}(A)$ has magnitude at most 1, we have that $\sum_{|A|\leqslant k} |\hat{C}_{\mathcal{D}}(A)|$ is at most n^k, and thus $|E_{\mathcal{D}}[f\chi_A]| \geqslant 1/(2sn^k)$ for some $|A| \leqslant k$. Thus we have proved the following lemma:

Lemma 6 *There is a subset $A \subset [n]$ with $|A| \leqslant k$ such that $|\Pr_{\mathcal{D}}[f(x) = \chi_A(x)] - \frac{1}{2}| \geqslant 1/(4sn^k)$ where $k = (20\log(8s^3 2^n L_\infty(\mathcal{D})))^d$.*

We can now establish the main result of this section:

Theorem 7 *Let $C_{s,d}$ be the class of MAC^0 circuits of size s and depth $d+1$ over $\{-1,1\}^n$. Let $\mathcal{D}$ be any distribution and let $k = (20\log(8s^3 2^n L_\infty(\mathcal{D})))^d$. There is a weak learning algorithm for $C_{s,d}$ with advantage $\gamma = 1/(8sn^k)$ which runs in time poly$(n^k, \log 1/\delta)$. The algorithm takes δ and γ as input and outputs a hypothesis which is either χ_A or $-\chi_A$ for some $|A| \leqslant k$.*

Proof: Lemma 6 implies that for some subset A of size at most k either χ_A or $-\chi_A$ will be the desired weak hypothesis. The weak learning algorithm does an exhaustive search over all $A \subset [n]$ in order of increasing size. For each candidate parity function χ_A the algorithm draws a sample of labeled examples from $EX(f,\mathcal{D})$ and uses this sample to compute an empirical estimate of $\Pr_{\mathcal{D}}[f(x) = \chi_A(x)]$. Suppose that the candidate parity being tested is $\chi_{A'}$. Using Chernoff bounds it can be shown that if a sample of size poly$(1/\gamma, \log 1/\delta')$ is used then with probability at least $1-\delta'$ the empirical estimate of $\Pr_{\mathcal{D}}[f(x) = \chi_{A'}(x)]$ will differ from the true value by at most $\gamma/2 = 1/(16sn^k)$. The algorithm halts and outputs $\chi_{A'}$ (or $-\chi_{A'}$) if and only if the empirical estimate p satisfies $|p-1/2| \geqslant 3\gamma/2 = 3/(16sn^k)$. It is easily verified that if δ' is taken to be $\delta/2^n$ then with probability at least $1-\delta$ this algorithm runs in time poly$(sn^k, \log(2^n/\delta))$ = poly$(n^k, \log 1/\delta)$ and succeeds as stated in the theorem. ■

3.2 Learning MAC^0 under the uniform distribution

To obtain a strong learning algorithm we will use the following theorem whose proof can be found in [20, 26]:[2]

Theorem 8 *There exists a canonical boosting algorithm* `B` *which, given input parameters $0 < \epsilon, \delta < 1$, $0 < \gamma < 1/2$, an example oracle $EX(f,\mathcal{U})$ and a weak learning algorithm* `WL` *with advantage γ, has the following properties:*

- `B` *runs for $T = O(1/\gamma^2\epsilon^2)$ stages;*
- *Each distribution $\mathcal{D}_i$ generated by* `B` *satisfies $L_\infty(\mathcal{D}_i) = O((1/\epsilon)2^{-n})$;*

[2] Similar results can be obtained from other boosting algorithms such as [10, 12, 14]. Each of these boosting algorithms gives slightly different bounds on T and $L_\infty(\mathcal{D}_i)$ but these differences in the bounds do not affect the running time of our overall algorithm.

- *The running time of* B *is polynomial in* $1/\gamma$, $1/\epsilon$, $\log 1/\delta$, n, *and* t, *where* t *is the running time of* WL *given access to* $EX(f, \mathcal{U})$;

- *The final hypothesis output by* B *is a majority* $h = MAJ(h_1, \ldots, h_{T-1})$ *of the hypotheses* h_i *generated by* WL.

We use this boosting algorithm in conjunction with the weak learning algorithm of Section 3.1 to construct a quasipolynomial time uniform distribution learning algorithm for MAC^0. Let $\epsilon > 0$ be the desired accuracy and confidence to which we will learn the MAC^0 circuit of size s and depth $d+1$. Every distribution $\mathcal{D}_i$ constructed by B satisfies $L_\infty(\mathcal{D}_i) = O((1/\epsilon)2^{-n})$. For each such distribution $\mathcal{D}_i$ Theorem 7 implies that our weak learning algorithm runs in time $\text{poly}(n^k, \log(1/\delta))$ and outputs a hypothesis with advantage $\gamma = 1/(8sn^k)$ where $k = O(\log^d(s/\epsilon))$. Theorem 8 implies that after $O(1/\gamma^2\epsilon^2)$ iterations with probability $1-\delta$ B outputs a hypothesis h such that $\Pr_{\mathcal{U}}[f(x) = h(x)] \geqslant 1 - \epsilon$. This hypothesis is a majority of $O(1/\gamma^2\epsilon^2)$ parity functions, each of size at most k.

Note that although we assumed that the value of γ is known to the boosting algorithm, this assumption can be removed by using a standard "guess-and-double" technique. The idea is to start with a constant-value guess for γ (e.g. $\gamma = 1/4$) and then run the algorithm with this value. If the algorithm fails to construct a strong hypothesis (which is easily tested), the value of γ is halved and the algorithm is run again. A standard analysis which we omit shows that this process incurs at most a polynomial increase in the run-time bound and hypothesis size. Thus the overall learning algorithm does not need to be given size or depth bounds of the MAC^0 circuit to be learned.

Summarizing the above discussion, we have the following theorem:

Theorem 9 *Let* $C_{s,d}$ *be the class of* MAC^0 *circuits of size* s *and depth* $d+1$ *over* $\{-1,1\}^n$. *The above algorithm learns* $C_{s,d}$ *to accuracy* ϵ *under the uniform distribution in time poly*$(n^{\log^d(s/\epsilon)}, \log 1/\delta)$.[3]

3.3 Learning constant-depth circuits with majority gates

Using the following result of Beigel [3] we can extend our learning result for MAC^0 to circuits which contain several majority gates:

Theorem 10 *Let* f *be computed by a circuit of size* s *and depth* d *which contains* m *majority gates. Then* f *is computed by a circuit of size* $2^{m(O(\log s))^{2d+1}}$ *and depth* $d+2$ *which contains a single majority gate at the root.*

Combining Theorems 9 and 10 we obtain our main learning result:

Theorem 11 *The class of size-*s *depth-*d *circuits which contain* m *majority gates can be learned to accuracy* ϵ *under the uniform distribution in time* $n^{O((m(\log s)^{2d+1} + \log(1/\epsilon))^{d+1})}$.

The most interesting setting of parameters in Theorem 11 is $s = 2^{polylog\, n}$, $d = O(1)$ and $m = \text{polylog}(n)$ which gives Theorem 1 as stated in the introduction.

[3] The analysis of the running time dependence on the confidence parameter δ is standard and omitted for clarity.

3.4 Discussion

Theorem 9 can be used to give a slight improvement on results of Linial *et al.* for learning and approximating constant-depth circuits. Since a majority gate can easily simulate an AND or OR gate by padding it with extra inputs, for any size-s depth-d AND/OR/NOT circuit C it is possible to reexpress C as an MAC^0 circuit $MAJ(C_1, \ldots, C_t)$ in which each C_i has size at most s and depth at most $d-1$. Applying Theorem 9 we see that any size-s depth-d AND/OR/NOT circuit can be learned using our algorithm in time $n^{O(\log^{d-1}(s/\epsilon))}$. This is in contrast with the Linial *et al.* algorithm which runs in time $n^{O(\log^{d}(s/\epsilon))}$.

By viewing AND/OR/NOT circuits as MAC^0 circuits in this way, our algorithm also gives the best known upper bound on the degree of polynomial threshold functions which approximate constant-depth circuits. A polynomial threshold function is a Boolean function defined by a real-valued multivariate polynomial $P(x)$. The Boolean function takes value 1 on x if and only if $P(x) \geqslant 0$.

As mentioned in Section 3.2, the final hypothesis of our algorithm is a majority of parities of at most k variables each. Since any Boolean function on k variables can be computed by a polynomial of degree at most k, we have the following corollary:

Corollary 12 *Let f be computed by a circuit of size s and depth d. Then there is a polynomial threshold function of degree $O(\log^{d-1}(s/\epsilon))$ which agrees with f on all but an ϵ fraction of inputs.*

The Linial *et al.* results imply the existence of a polynomial threshold function approximator of degree $O(\log^{d}(s/\epsilon))$.

Aspnes *et al.* [1] have given tight bounds on the accuracy with which polynomial threshold functions of a given degree can approximate the parity function. Using their results together with the fact that a depth-d circuit of size s can compute parity on $\log^{d-1} s$ variables, one can show

Observation 13 *There is a circuit C of size s and depth d such that any polynomial threshold function of degree $O(\log^{d/2} s)$ must disagree with C on more than an $\epsilon = 1/s$ fraction of inputs.*

Thus the $d-1$ exponent in our upper bound of Corollary 12 is worse than the best possible $(d/2)$ exponent by less than a factor of two.

Finally, we observe that as a corollary of Theorem 11 we obtain a quasipolynomial time algorithm for learning an intersection of $\mathrm{polylog}(n)$ halfspaces under the uniform distribution on the Boolean cube, under the restriction that each halfspace is defined by integer coefficients of magnitude $\mathrm{poly}(n)$. This is in contrast with a polynomial time algorithm by Baum [2] for learning an intersection of two origin-centered halfspaces under a restricted class of distributions which includes the uniform distribution. (Vempala [29] gave a polynomial-time algorithm to learn an intersection of $\log n/\log\log n$ halfspaces under "near-uniform" distributions on the unit ball, but his algorithm does not apply to the uniform distribution on the Boolean cube.)

4 Cryptographic Lower Bounds

In contrast with the positive learning results of Section 3, in this section we give cryptographic hardness results for learning circuits with majority gates. Our results, which are based on the presumed intractability of factoring Blum integers, suggest that the time bounds of Theorem 11 are essentially optimal even for weak learning algorithms which are allowed to make membership queries.

Definition 14 *Let $N = P \cdot Q$ be chosen by uniformly selecting P and Q to be two $n/2$-bit primes such that $P \equiv Q \equiv 3 \bmod 4$ (we call such an N a* random n-bit Blum integer*). Let $\epsilon(n)$ and $t(n)$ be real-valued functions. We say that a probabilistic $t(n)$-time algorithm A* ϵ-factors *if*

$$\Pr[A(N) \in \{P, Q\}] > \epsilon(n)$$

where $N = P \cdot Q$ is a random n-bit Blum integer. The $(t(n), \epsilon(n))$-factoring assumption *states that there is no probabilistic poly($t(n)$)-time algorithm which poly($\epsilon(n)$)-factors.*

After decades of intensive research on integer factorization, the fastest factoring algorithms known for n-bit Blum integers run in time $2^{\tilde{O}(n^{1/3})}$. Thus the following assumption is viewed as being quite plausible [19]: there is some absolute constant $\alpha > 0$ such that the $(2^{n^\alpha}, 2^{-n^\alpha})$-factoring assumption holds.

Definition 15 *Let $I(n)$ be an index set and let $\{f_i : \{-1,1\}^n \to \{-1,1\}\}_{i \in I(n)}$ be a family of functions. We say that $\{f_i\}_{i \in I(n)}$ is an* $(t(n), \epsilon(n))$-secure pseudorandom function family *if the following conditions hold:*

- *there is a probabilistic poly(n)-time algorithm which selects a uniform random $i \in I(n)$;*
- *there is a deterministic poly(n)-time algorithm which given $i \in I(n)$ and $x \in \{-1,1\}^n$ outputs $f_i(x)$;*
- *for any probabilistic $t(n)$-time oracle algorithm A,*

$$\left| \Pr_{i \in I(n)}[A^{f_i} \text{ outputs } 1] - \Pr_{F \in \mathcal{F}_n}[A^F \text{ outputs } 1] \right| < \epsilon(n)$$

  where $\mathcal{F}_n$ is the family of all Boolean functions $F : \{0,1\}^n \to \{0,1\}$.

The following theorem follows from [25]:

Theorem 16 *There is an index set $I(n)$, a function family $\{f_i : \{0,1\}^n \to \{-1,1\}\}_{i \in I(n)}$, and a fixed constant $\beta > 0$ such that*

- *Each function $f_i : \{-1,1\}^n \to \{-1,1\}$ is computed by a depth-5 circuit consisting of n^β majority gates;*
- *If $\{f_i\}_{i \in I(n)}$ is not a $(t(n), \epsilon(n))$-secure pseudorandom function family then there is a probabilistic poly($t(n)/\epsilon(n)$)-time algorithm which poly($\epsilon(n)/t(n)$)-factors.*

The following theorem about hardness of learning follows:

Theorem 17 *Suppose that there is a membership query learning algorithm A for the class $\{f_i : \{-1,1\}^n \to \{-1,1\}\}_{i \in I(n)}$ of Theorem 16 which runs in time $t(n)$ and outputs a hypothesis h such that $\Pr[h(x) \neq f(x)] \leqslant \frac{1}{2} - \epsilon(n)$. Then the $(t'(n), \epsilon'(n))$-factoring assumption is false for some $t'(n)$ =poly($t(n)/\epsilon(n)$) and $\epsilon'(n)$ =poly($\epsilon(n)/t(n)$).*

We can apply Theorem 17 to obtain hardness results for learning constant-depth circuits with majority gates:

Theorem 18 *Suppose that the $(2^{n^\alpha}, 2^{-n^\alpha})$-factoring assumption holds for some fixed $\alpha > 0$. Then any membership query algorithm that weakly learns depth 5 circuits consisting of m majority gates to accuracy $\epsilon = 1/2 - 2^{-m^{\alpha/\beta}}$ must have time complexity $\omega(2^{m^{\alpha/\beta}})$, where β is the constant from Theorem 16.*

Proof: From Theorem 16 we know that every function $f_{i,m^{1/\beta}}$ can be computed by a depth 5 circuit consisting of m majority gates. If A runs in time $2^{m^{\alpha/\beta}}$ and weakly learns $\{f_{i,m^{1/\beta}}\}$ to accuracy $1/2 - 2^{-m^{\alpha/\beta}}$, then A can weakly learn $\{f_{i,n}\}$ to accuracy $1/2 - 2^{-n^\alpha}$ in time 2^{n^α}. The result follows by Theorem 17. ■

As a corollary we have that our algorithm for learning constant-depth circuits with polylogarithmically many majority gates is essentially optimal in terms of its time complexity and the number of majority gates which it can handle.

Corollary 19 *Fix any constants $d \geqslant 5$ and $\epsilon > 0$.*

1. *Our learning algorithm from Section 3 learns depth-d poly(n)-size circuits with $\log^k n$ majority gates to accuracy ϵ in time $2^{(\log n)^{O(k)}}$.*
2. *Assuming that the $(2^{n^\alpha}, 2^{-n^\alpha})$-factoring assumption holds for some fixed $\alpha > 0$, any algorithm (even using membership queries) which learns depth-d circuits consisting of $\log^k n$ majority gates (even to accuracy $1/2 - 2^{-(\log n)^{\Theta(k)}}$) must run in time $2^{(\log n)^{\Omega(k)}}$.*
3. *Assuming that the $(2^{n^\alpha}, 2^{-n^\alpha})$-factoring assumption holds for some fixed $\alpha > 0$, then for any $m = \omega(polylog(n))$ there is no quasipolynomial-time algorithm (even using membership queries) which learns depth-d circuits consisting of m majority gates (even to accuracy $1/2 - 2^{-m^{\Theta(1)}}$).*

5 Learning other extensions of AC^0

Given our results for learning AC^0 augmented with majority gates, a natural question is whether AC^0 circuits augmented with other expressive gates can be similarly learned. A first step in this direction is to investigate the learnability of constant-depth polynomial-size circuits with a single parity gate, MOD_p gate, or threshold gate at the root. We refer to these three circuit classes as *parity of* AC^0, MOD_p *of* AC^0 and *threshold of* AC^0. While we have not been able to prove cryptographic hardness results for these three classes, as described below we believe that substantially new techniques will be required to obtain learning algorithms for these classes.

The most powerful algorithms known for learning Boolean circuits under the uniform distribution are based on identifying large Fourier coefficients. Indeed, the only classes for which uniform distribution quasipolynomial time learning algorithms are known are classes such that for each $f \in C$ there is some Fourier coefficient such that $|\hat{f}_A| \geqslant 1/n^{\mathrm{polylog}(n)}$. (Our analysis in Section 3 implies the existence of such a Fourier coefficient for any constant-depth circuit with polylog(n) majority gates.) For parity of AC^0 it can be shown that all Fourier coefficients may be exponentially small. This is almost true of MOD_p of AC^0 as well: the only exception is the constant coefficient, and we argue that existing methods are not powerful enough to produce anything but weak learning of such functions.[4] Thus it appears that Fourier techniques will not suffice to learn parity of AC^0 or MOD_p of AC^0.

[4]Recall that under the uniform distribution weak learning is not always equivalent to strong learning, e.g. the class of monotone Boolean functions is weakly but not strongly learnable under uniform [18].

For threshold of AC^0, we give evidence that our algorithm from Section 3 will not work by constructing a threshold of AC^0 function with the following property: under the uniform distribution on $\{-1,1\}^n$ each input to the threshold gate has only an exponentially small correlation with the output of the circuit. (Recall that our algorithm's proof of correctness relies on the fact that for a majority of AC^0 circuit there is always some input to the majority gate which has nonnegligible correlation with the output.)

We defer these proofs and constructions to the appendix.

References

[1] J. Aspnes, R. Beigel, M. Furst, and S. Rudich. The expressive power of voting polynomials. *Combinatorica*, 14(2):1–14, 1994.

[2] E. Baum. A polynomial time algorithm that learns two hidden unit nets. *Neural Computation*, 2:510–522, 1991.

[3] R. Beigel. When do extra majority gates help? polylog(n) majority gates are equivalent to one. *Computational Complexity*, 4:314–324, 1994.

[4] R. Beigel, N. Reingold, and D. Spielman. Pp is closed under intersection. *Journal of Computer and System Sciences*, 50(2):191–202, 1995.

[5] A. Blum. Rank-r decision trees are a subclass of r-decision lists. *Information Processing Letters*, 42(4):183–185, 1992.

[6] A. Blum, A. Frieze, R. Kannan, and S. Vempala. A polynomial time algorithm for learning noisy linear threshold functions. *Algorithmica*, 22(1/2):35–52, 1997.

[7] A. Blumer, A. Ehrenfeucht, D. Haussler, and M. Warmuth. Learnability and the vapnik-chervonenkis dimension. *Journal of the ACM*, 36(4):929–965, 1989.

[8] J. Bruck. Harmonic analysis of polynomial threshold functions. *SIAM Journal on Discrete Mathematics*, 3(2):168–177, 1990.

[9] N. Bshouty. A subexponential exact learning algorithm for dnf using equivalence queries. *Information Processing Letters*, 59:37–39, 1996.

[10] N. Bshouty and D. Gavinsky. On boosting with optimal poly-bounded distributions. In *Proceedings of the Fourteenth Annual Conference on Computational Learning Theory / Fifth European Conference on Computational Learning Theory*, pages 490–506, 2001.

[11] N. Bshouty and C. Tamon. On the fourier spectrum of monotone functions. *Journal of the ACM*, 43(4):747–770, 1996.

[12] C. Domingo and O. Watanabe. Madaboost: a modified version of adaboost. In *Proceedings of the Thirteenth Annual Conference on Computational Learning Theory*, pages 180–189, 2000.

[13] A. Ehrenfeucht and D. Haussler. Learning decision trees from random examples. *Information and Computation*, 82(3):231–246, 1989.

[14] Y. Freund. Boosting a weak learning algorithm by majority. In *Proceedings of the Third Annual Workshop on Computational Learning Theory*, pages 202–216, 1990.

[15] A. Hajnal, W. Maass, P. Pudlak, M. Szegedy, and G. Turan. Threshold circuits of bounded depth. *Journal of Computer and System Sciences*, 46:129–154, 1993.

[16] J. Hastad. *Computational Limitations for Small Depth Circuits.* MIT Press, Cambridge, MA, 1986.

[17] J. Jackson. An efficient membership-query algorithm for learning dnf with respect to the uniform distribution. *Journal of Computer and System Sciences*, 55:414–440, 1997.

[18] M. Kearns, M. Li, and L. Valiant. Learning boolean formulas. *Journal of the ACM*, 41(6):1298–1328, 1994.

[19] M. Kharitonov. Cryptographic hardness of distribution-specific learning. In *Proceedings of the Twenty-Fifth Annual Symposium on Theory of Computing*, pages 372–381, 1993.

[20] A. Klivans and R. Servedio. Boosting and hard-core sets. In *Proceedings of the Fortieth Annual Symposium on Foundations of Computer Science*, pages 624–633, 1999.

[21] A. Klivans and R. Servedio. Learning dnf in time $2^{\tilde{o}(n^{1/3})}$. In *Proceedings of the Thirty-Third Annual Symposium on Theory of Computing*, pages 258–265, 2001.

[22] M. Krause and P. Pudlak. On the computational power of depth 2 circuits with threshold and modulo gates. In *Proceedings of the Twenty-Sixth Annual Symposium on Theory of Computing*, pages 48–57, 1994.

[23] N. Linial, Y. Mansour, and N. Nisan. Constant depth circuits, fourier transform and learnability. *Journal of the ACM*, 40(3):607–620, 1993.

[24] F.J. MacWilliams and N.J.A. Sloane. *The Theory of Error-Correcting Codes.* North-Holland, New York, 1977.

[25] M. Naor and O. Reingold. Number-theoretic constructions of efficient pseudo-random functions. In *Proceedings of the Thirty-Eighth Annual Symposium on Foundations of Computer Science*, pages 458–467, 1997.

[26] R. Servedio. *Efficient Algorithms in Computational Learning Theory.* PhD thesis, Harvard University, 2001.

[27] J. Tarui and T. Tsukiji. Learning dnf by approximating inclusion-exclusion formulae. In *Proceedings of the Fourteenth Conference on Computational Complexity*, pages 215–220, 1999.

[28] L. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

[29] S. Vempala. A random sampling based algorithm for learning the intersection of halfspaces. In *Proceedings of the 38th Annual Symposium on Foundations of Computer Science*, pages 508–513, 1997.

[30] Z. L. Zhang. Complexity of symmetric functions in perceptron-like models. Master's thesis, University of Massachusetts at Amherst, 1992.

A Bounds on the Fourier coefficients of Extensions of AC^0

A.1 Parity of AC^0

The *inner product mod 2* function $IP_2 : \{0,1\}^{2n} \rightarrow \{0,1\}$ is defined by $IP_2(x) = x_1x_2 + x_3x_4 + \ldots + x_{2n-1}x_{2n}$ mod 2. This function is computed by a depth 2 circuit with n AND gates on the bottom level which feed into a single parity gate at the root. It is known [8, 24] that every Fourier coefficient of this function satisfies $|\widehat{IP}_2(S)| = 1/2^n$.

We note that the IP_2 function can also be used to show (without any cryptographic assumptions) that our learning algorithm from Section 3 will not run in quasipolynomial time on depth-4 circuits with more than a polylogarithmic number of majority gates. Using majority gates to compute parity, the IP_2 function on m variables can be computed by a depth-4 size-m^2 circuit which contains m majority gates. Let $f_m : \{0,1\}^n \rightarrow \{0,1\}$ denote the inner product mod 2 function on the first m variables $x_1, \ldots, x_m$. For $m = \omega(\text{polylog}(n))$, the first execution of our weak learning algorithm (under the uniform distribution) will take more than quasipolynomial time to identify a parity function which is weakly correlated with the target function.

A.2 MOD_p of AC^0 for $p > 2$

For $x \in \{0,1\}^n$ the function $MOD_p(x)$ outputs 1 if and only if $x_1 + \ldots + x_n$ is an integer multiple of p. It has been shown by Zhang [30] that for any fixed prime $p > 2$ the function $MOD_p(x)$ has $|\widehat{MOD}_p(S)| = 1/2^{\Omega(n)}$ for all $S \neq \varnothing$. This observation taken by itself leaves open the possibility that boosting and Fourier methods could be combined to learn this class as follows: choose the constant function $-\chi_\varnothing$ as the first weak hypothesis. Then apply boosting to shift the distribution. Perhaps some other parity function will now be a weak approximator with respect to this new distribution and can be found reasonably efficiently. If so, then potentially a strong hypothesis can be produced by continuing this boosting process.

However, we argue that this approach cannot succeed given current boosting algorithms. Every known boosting algorithm that is given a constant function as the first weak hypothesis will form the distribution for the next boosting round in a very simple way: it gives more weight to all positive examples and less weight to all negative examples, or vice versa. More precisely, it will change the weight on all positive examples by the same additive or multiplicative factor, and it will similarly change the weight on all negative examples by using a single factor. If the initial distribution has at most two distinct weights, one shared by all positive examples and the other shared by all negative examples, then the new distribution will have the same form—all positives share a single weight, all negatives a different weight—but will have weights that differ from the first distribution.

Also, while the specific weight changes may vary with the boosting algorithm, in no case will the change be greater than is required to balance the weight on the positive and negative examples. Specifically, in the case of learning MOD_p with respect to the uniform distribution, the most weight put on examples in the second boosting distribution $\mathcal{D}_1$ will be $(p/2)/2^n$.

We will show that after any such change to the distribution it will still be the case that only the constant parity function has more than inverse exponential correlation with the target MOD_p function with respect to the uniform distribution.

So consider the correlation of any parity function χ_B, $B \neq \varnothing$, with the target MOD_p function f with respect to $\mathcal{D}_1$. Let $w_{-1}/2^n$ be the weight placed on each negative example by $\mathcal{D}_1$ and $w_1/2^n$

be the weight on each positive example. Then

$$E_{\mathcal{D}_1}[f\chi_B] = \frac{w_{-1}}{2^n} \sum_{x.f(x)=-1} -\chi_B(x) + \frac{w_1}{2^n} \sum_{x.f(x)=1} \chi_B(x).$$

Now let $f'(x) = (f(x)+1)/2$ (f' is the 0/1-valued function that is 1 if and only if f is 1). Then for all $A \neq \varnothing$, $\widehat{f'}(A) = \hat{f}(A)/2$. Similarly, for $f''(x) = (1 - f(x))/2$ (1 if and only if f is -1), $\widehat{f''}(A) = -\hat{f}(A)/2$ for all $A \neq \varnothing$. So

$$E_{\mathcal{D}_1}[f\chi_B] = w_1 E_{\mathcal{U}}[f'(x)\chi_B(x)] - w_{-1} E_{\mathcal{U}}[f''(x)\chi_B(x)] = \frac{w_{-1}+w_1}{2}\hat{f}(B).$$

Since the w's are at most linear in p, and since $\hat{f}(B)$ is exponentially small, we have that every parity function (except possibly the constant) is very weakly correlated with f with respect to $\mathcal{D}_1$. This implies that at the next boosting step, any known Fourier-based learning strategy can do no better than choosing the constant as the weak hypothesis. But this will simply lead to a distribution $\mathcal{D}_2$ of the same form as $\mathcal{D}_1$ and with the same constraints on the weights.

B Learning Threshold of AC^0

A threshold function $t : \{-1,1\}^n \to \{-1,1\}$ is defined by real numbers $w_1, \ldots, w_n, \theta$; the value of $t(x)$ is 1 if and only if $w_1x_1 + \cdots + w_nx_n \geqslant \theta$. We do not know whether every threshold of AC^0 function must have some Fourier coefficient of inverse quasipolynomial size. Thus it is conceivable that our algorithm in Section 3 may be able to learn any threshold of AC^0 function in quasipolynomial time. However, we can show that our analysis given in Section 3.1 breaks down for threshold of AC^0. More specifically we construct a threshold of AC^0 function such that even under the uniform distribution on $\{-1,1\}^n$ no input to the threshold gate is nonnegligibly correlated with its output. We use the following lemma (whose proof is omitted from this extended abstract):

Lemma 20 *Let $n \geqslant 1$ be odd and let $t : \{-1,1\}^n \to \{-1,1\}$ be the threshold function defined by $t(x) \stackrel{def}{=} (-1)^{(j+1)}$ where $1 \leqslant j \leqslant n$ is the smallest index such that $x_j = 1$ (if $x_j = -1$ for all j then $t(x) = -1$). For $1 \leqslant i \leqslant n+1$ and $1 \leqslant j \leqslant n$ let $x^i \in \{0,1\}^n$ be defined as follows:*

$$x^i_j = \begin{cases} -1 & \text{if } i > j \\ 1 & \text{if } i = j \\ (-1)^{i+j+1} & \text{if } i < j. \end{cases}$$

Let F_ℓ denote the ℓ-th Fibonacci number (so $F_1 = F_2 = 1, F_3 = 2$, etc.) and let $G_\ell \stackrel{def}{=} F_{\ell+2} + F_{\ell-1} - 1$. Let $\mathcal{D}$ be the following distribution over $\{-1,1\}^n$:

$$\mathcal{D}(x) = \begin{cases} F_i/G_n & \text{if } x = x^i \text{ for some } 1 \leqslant i \leqslant n \\ F_{n-1}/G_n & \text{if } x = x^{n+1} \\ 0 & \text{otherwise.} \end{cases}$$

Then for each $i = 1, \ldots, n$ we have $|E_{\mathcal{D}}[x_i t(x)]| = 1/G_n = 1/2^{\Omega(n)}$ and moreover $E_{\mathcal{D}}[t(x)] = 1/G_n = 1/2^{\Omega(n)}$.

As the first step of the analysis in Section 3.1, we observed that for any distribution $\mathcal{D}$ and any MAC^0 circuit $f = MAJ(C_1, \ldots, C_t)$ there is some AC^0 circuit C_i which is an input to the top-level majority gate such that $|E_{\mathcal{D}}[C_i f]| \geqslant 1/t$. We now show that this property does not hold for threshold of AC^0 even if the distribution is uniform. More precisely, we have

Lemma 21 *There is a threshold of* AC^0 *function* $g : \{-1,1\}^n \to \{-1,1\}$, *defined by* $g(x) = t(C_1, \ldots, C_n)$ *where each* C_i *is a polynomial-size depth-2 circuit, such that for* $1 \leqslant i \leqslant n$ *we have* $|E[C_i g]| = O(1/G_n)$ *and moreover* $E[g] = O(1/G_n)$.

Proof: We exhibit constant-depth circuits $C_1, \ldots, C_n$ with the following property: given a uniformly distributed input $x \in \{-1,1\}^n$, the distribution of the n-bit string $C_1(x)C_2(x)\ldots C_n(x)$ which is input to t is exponentially close to the distribution $\mathcal{D}$ described in Lemma 20. This is done as follows: let $[0, a_1], [a_1+1, a_2], \ldots, [a_{n-1}+1, a_n], [a_n+1, 2^n-1]$ be a partition of $\{0, 1, \ldots, 2^n-1\}$ into $n+1$ disjoint intervals where the i-th interval $[a_{i-1}+1, a_i]$ is of length approximately $2^n\mathcal{D}(x^i)$. On input $x \in \{-1,1\}^n$ the j-th circuit C_j outputs 1 if and only if the number in $\{0, 1, \ldots, 2^n-1\}$ corresponding to x lies in an interval $[a_{k-1}+1, a_k]$ such that the i-th bit of x^k is 1.

It is easily verified that the above choice of $a_1, \ldots, a_n$ causes the distribution of $C_1(x)C_2(x)\ldots C_n(x)$ to be exponentially close to $\mathcal{D}$. Each circuit C_i as described above must output 1 if and only if its input, viewed as an integer, lies in a union of at most n intervals in $\{0, 1, \ldots, 2^n-1\}$. Such circuits can be constructed in polynomial size and depth 2. ■

We conclude by observing that known polynomial-time algorithms for learning a single threshold gate make essential use of geometric techniques (e.g. the linear programming algorithms used by Blumer *et al.* [7] and the "outlier removal lemma" of Blum *et al.* [6]). These techniques have a very different feel from the Fourier methods which have proved useful for uniform distribution learning. It would be interesting to have a learning algorithm which combines these two approaches.