BQP and the Polynomial Hierarchy

The MIT Faculty has made this article openly available. Please share how this access benefits you. Your story matters.

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>As Published</td>
<td></td>
</tr>
<tr>
<td>Publisher</td>
<td>Association for Computing Machinery</td>
</tr>
<tr>
<td>Version</td>
<td>Original manuscript</td>
</tr>
<tr>
<td>Accessed</td>
<td>Wed Mar 16 14:18:56 EDT 2016</td>
</tr>
<tr>
<td>Citable Link</td>
<td><a href="http://hdl.handle.net/1721.1/54236">http://hdl.handle.net/1721.1/54236</a></td>
</tr>
<tr>
<td>Terms of Use</td>
<td>Creative Commons Attribution-Noncommercial-Share Alike 3.0</td>
</tr>
<tr>
<td>Detailed Terms</td>
<td><a href="http://creativecommons.org/licenses/by-nc-sa/3.0/">http://creativecommons.org/licenses/by-nc-sa/3.0/</a></td>
</tr>
</tbody>
</table>
BQP and the Polynomial Hierarchy

Scott Aaronson

Abstract

The relationship between \textit{BQP} and \textit{PH} has been an open problem since the earliest days of quantum computing. We present evidence that quantum computers can solve problems outside the entire polynomial hierarchy, by relating this question to topics in circuit complexity, pseudorandomness, and Fourier analysis.

First, we show that there exists an oracle relation problem (i.e., a problem with many valid outputs) that is solvable in \textit{BQP}, but not in \textit{PH}. This also yields a non-oracle relation problem that is solvable in quantum logarithmic time, but not in \textit{AC}^0.

Second, we show that an oracle decision problem separating \textit{BQP} from \textit{PH} would follow from the \textit{Generalized Linial-Nisan Conjecture}, which we formulate here and which is likely of independent interest. The original Linial-Nisan Conjecture (about pseudorandomness against constant-depth circuits) was recently proved by Braverman, after being open for twenty years.

Contents

1 Introduction ................................................. 2
   1.1 Motivation .............................................. 3
   1.2 Our Results ............................................ 4
   1.3 In Defense of Oracles ................................... 8

2 Preliminaries .............................................. 8
   2.1 Problems ............................................... 8
   2.2 Complexity Classes .................................... 10

3 Quantum Algorithms ...................................... 11
   3.1 Quantum Algorithm for \textit{Fourier Fishing} ............ 12
   3.2 Quantum Algorithm for \textit{Fourier Checking} ........... 14

4 The Classical Complexity of \textit{Fourier Fishing} ......... 16
   4.1 Constant-Depth Circuit Lower Bounds .................... 16
   4.2 Secretly Biased Fourier Coefficients .................... 18
   4.3 Putting It All Together ................................ 20

5 The Classical Complexity of \textit{Fourier Checking} ....... 23
   5.1 Almost \(k\)-Wise Independence .......................... 24
   5.2 Oracle Separation Results ............................. 28

*MIT. Email: aaronson@csail.mit.edu. This material is based upon work supported by the National Science Foundation under Grant No. 0844626. Also supported by a DARPA YFA grant and the Keck Foundation.
1 Introduction

A central task of quantum computing theory is to understand how BQP—meaning Bounded-Error Quantum Polynomial-Time, the class of all problems feasible for a quantum computer—fits in with classical complexity classes. In their original 1993 paper defining BQP, Bernstein and Vazirani [13] showed that BPP ⊆ BQP ⊆ P^#P.\(^1\) Informally, this says that quantum computers are at least as fast as classical probabilistic computers and no more than exponentially faster (indeed, they can be simulated using an oracle for counting). Bernstein and Vazirani also gave evidence that BPP ≠ BQP, by exhibiting an oracle problem called RECURSIVE FOURIER SAMPLING that requires \(n^{\Omega(\log n)}\) queries on a classical computer but only \(n\) queries on a quantum computer.\(^2\) The evidence for the power of quantum computers became dramatically stronger a year later, when Shor [37] (building on work of Simon [38]) showed that FACTORING and DISCRETE LOGARITHM are in BQP. On the other hand, Bennett et al. [11] gave oracle evidence that NP ⊄ BQP, and while no one regards such evidence as decisive, today it seems extremely unlikely that quantum computers can solve NP-complete problems in polynomial time. A vast body of research, continuing to the present, has sought to map out the detailed boundary between those NP problems that are feasible for quantum computers and those that are not.

However, there is a complementary question that—despite being universally recognized as one of the “grand challenges” of the field—has had essentially zero progress over the last sixteen years:

\[\text{Is BQP in NP? More generally, is BQP contained anywhere in the polynomial hierarchy PH = NP U NP^{NP} U NP^{NP^{NP}} U \cdots ?}\]

The “default” conjecture is presumably BQP ⊄ PH, since no one knows what a simulation of BQP in PH would look like. Before this work, however, there was no formal evidence for or against that conjecture. Almost all the problems for which we have quantum algorithms—including FACTORING and DISCRETE LOGARITHM—are easily seen to be in NP \(\cap\) coNP.\(^3\) One notable exception is RECURSIVE FOURIER SAMPLING, the problem that Bernstein and Vazirani [13] originally used to construct an oracle \(A\) relative to which BPP\(^A\) ≠ BQP\(^A\). One can show, without too much difficulty, that RECURSIVE FOURIER SAMPLING yields oracles \(A\) relative to which BQP\(^A\) ⊄ NP\(^A\) and indeed BQP\(^A\) ⊄ MA\(^A\). However, while it is reasonable to conjecture that RECURSIVE FOURIER SAMPLING (as an oracle problem) is not in PH, it is open even to show that this problem (or any other BQP oracle problem) is not in AM! Recall that AM = NP under

---

\(^1\)The upper bound was later improved to BQP ⊆ PP by Adleman, DeMarrais, and Huang [4].

\(^2\)For more about RECURSIVE FOURIER SAMPLING see Aaronson [2].

\(^3\)Here we exclude BQP-complete problems such as approximating the Jones polynomial [6], which, by the very fact of being BQP-complete, seem hard to interpret as “evidence” for BQP ⊄ PH.
plausible derandomization assumptions [29]. Thus, until we solve the problem of constructing an oracle $A$ such that $BQP^A \not\subseteq AM^A$, we cannot even claim to have oracle evidence (which is itself, of course, a weak form of evidence) that $BQP \not\subseteq NP$.

Before going further, we should clarify that there are two questions here: whether $BQP \subseteq PH$ and whether $\text{PromiseBQP} \subseteq \text{PromisePH}$. In the unrelativized world, it is entirely possible that quantum computers can solve promise problems outside the polynomial hierarchy, but that all languages in $BQP$ are nevertheless in $PH$. However, for the specific purpose of constructing an oracle $A$ such that $BQP^A \not\subseteq PH^A$, the two questions are equivalent, basically because one can always “offload” a promise into the construction of the oracle $A$.\footnote{Here is a simple proof: let $\Pi = (\Pi_{YES}, \Pi_{NO})$ be a promise problem in $\text{PromiseBQP}^A \setminus \text{PromisePH}^A$, for some oracle $A$. Then clearly, every $\text{PromisePH}^A$ machine $M$ fails to solve $\Pi$ on infinitely many inputs $x$ in $\Pi_{YES} \cup \Pi_{NO}$. This means that we can produce an infinite sequence of inputs $x_1, x_2, \ldots$ in $\Pi_{YES} \cup \Pi_{NO}$, whose lengths $n_1, n_2, \ldots$ are spaced arbitrarily far apart, such that every $\text{PromisePH}^A$ machine $M$ fails to solve $\Pi$ on at least one $x_i$. Now let $B$ be an oracle that is identical to $A$, except that for each input length $n$, it reveals (i) whether $n = n_i$ for some $i$ and (ii) if so, what the corresponding $x_i$ is. Also, let $L$ be the unary language that contains $0^n$ if and only if (i) $n = n_i$ for some $i$ and (ii) $x_i \in \Pi_{YES}$. Then $L$ is in $BQP^B$ but not $PH^B$.}

1.1 Motivation

There are at least four reasons why the $BQP$ versus $PH$ question is so interesting. At a basic level, it is both theoretically and practically important to understand what classical resources are needed to simulate quantum physics. For example, when a quantum system evolves to a given state, is there always a short classical proof that it does so? Can one estimate quantum amplitudes using approximate counting (which would imply $BQP \subseteq \text{BPP}^NP$)? If something like this were true, then while the exponential speedup of Shor’s factoring algorithm might stand, quantum computing would nevertheless seem much less different from classical computing than previously thought.

Second, if $BQP \not\subseteq PH$, then many possibilities for new quantum algorithms might open up to us. One often hears the complaint that there are too few quantum algorithms, or that progress on quantum algorithms has slowed since the mid-1990s. In our opinion, the real issue here has nothing to do with quantum computing, and is simply that there are too few natural $NP$-intermediate problems for which there plausibly could be quantum algorithms! In other words, instead of focussing on $GRAPH$ ISOMORPHISM and a small number of other $NP$-intermediate problems, it might be fruitful to look for quantum algorithms solving completely different types of problems—problems that are not necessarily even in $PH$. In this paper, we will see a new example of such a quantum algorithm, which solves a problem called FOURIER CHECKING.

Third, it is natural to ask whether the $P \not\subseteq BQP$ question is related to that other fundamental question of complexity theory, $P \not\subseteq NP$. More concretely, is it possible that quantum computers could provide exponential speedups even if $P = NP$? If $BQP \subseteq PH$, then certainly the answer to that question is no (since $P = NP \implies P = PH$). Therefore, if we want evidence that quantum computing could survive a collapse of $P$ and $NP$, we must also seek evidence that $BQP \not\subseteq PH$.

Fourth, a major challenge for quantum computing research is to get better evidence that quantum computers cannot solve $NP$-complete problems in polynomial time. As an example, could we show that if $NP \subseteq BQP$, then the polynomial hierarchy collapses? At first glance, this seems like a wild hope; certainly we have no idea at present how to prove anything of the kind. However, notice
that if $\text{BQP} \subseteq \text{AM}$, then the desired implication would follow immediately! For in that case,

$$\text{NP} \subseteq \text{BQP} \implies \text{coNP} \subseteq \text{BQP}$$

$$\implies \text{coNP} \subseteq \text{AM}$$

$$\implies \text{PH} = \Sigma^P_2$$

where the last implication was shown by Boppana, H˚astad, and Zachos [14]. Similar remarks apply to the questions of whether $\text{NP} \subseteq \text{BQP}$ would imply $\text{PH} \subseteq \text{BQP}$, and whether the folklore result $\text{NP}^{\text{BPP}} \subseteq \text{BPP}^{\text{NP}}$ has the quantum analogue $\text{NP}^{\text{BQP}} \subseteq \text{BQP}^{\text{NP}}$. In each of these cases, we find that understanding some other issue in quantum complexity theory requires first coming to grips with whether $\text{BQP}$ is contained in some level of the polynomial hierarchy.

1.2 Our Results

This paper presents the first formal evidence for the possibility that $\text{BQP} \not\subset \text{PH}$. Perhaps more importantly, it places the relativized $\text{BQP}$ versus $\text{PH}$ question at the frontier of (classical) circuit lower bounds. The heart of the problem, we will find, is to extend Braverman’s spectacular recent proof [15] of the Linial-Nisan Conjecture, in ways that would reveal a great deal of information about small-depth circuits independent of the implications for quantum computing.

We have two main contributions. First, we achieve an oracle separation between $\text{BQP}$ and $\text{PH}$ for the case of relation problems. A relation problem is simply a problem where the desired output is an $n$-bit string (rather than a single bit), and any string from some nonempty set $S$ is acceptable. Relation problems arise often in theoretical computer science; one well-known example is finding a Nash equilibrium (shown to be $\text{PPAD}$-complete by Daskalakis et al. [18]). Within quantum computing, there is considerable precedent for studying relation problems as a warmup to the harder case of decision problems. For example, in 2004 Bar-Yossef, Jayram, and Kerenidis [7] gave a relation problem with quantum one-way communication complexity $O(\log n)$ and randomized one-way communication complexity $\Omega(\sqrt{n})$. It took several more years for Gavinsky et al. [23] to achieve the same separation for decision problems, and the proof was much more complicated. The same phenomenon has arisen many times in quantum communication complexity [20, 21, 22, 24, 25], though to our knowledge, this is the first time it has arisen in quantum query complexity.

Formally, our result is as follows:

**Theorem 1** There exists an oracle $A$ relative to which $\text{FBQP}^A \not\subset \text{FBPP}^{\text{PH}^A}$, where $\text{FBQP}$ and $\text{FBPP}$ are the relation versions of $\text{BQP}$ and $\text{BPP}$ respectively.

Interestingly, the oracle $A$ in Theorem 1 can be taken to be a random oracle. By contrast, Aaronson and Ambainis [3] have shown that, under a plausible conjecture about influences in low-degree polynomials, one cannot separate the decision classes $\text{BQP}^A$ and $\text{BPP}$ by a random oracle without also separating $\text{P}$ from $\text{P}^{\text{#P}}$ in the unrelativized world. In other words, the use of relation problems (rather than decision problems) seems essential if one wants a random oracle separation.

---

*Recently, Shepherd and Bremner [36] proposed a beautiful sampling problem that is solvable efficiently on a quantum computer but that they conjecture is hard classically. This provides yet another example where it seems easier to find evidence for the power of quantum computers once one moves away from decision problems.

*Confusingly, the $F$ stands for “function”; we are simply following the standard naming convention for classes of relation problems ($\text{FP}$, $\text{FNP}$, etc).
Underlying Theorem 1 is a new lower bound against \( \text{AC}^0 \) circuits (constant-depth circuits composed of AND, OR, and NOT gates). The close connection between \( \text{AC}^0 \) and the polynomial hierarchy that we exploit is not new. In the early 1980s, Furst-Saxe-Sipser [19] and Yao [43] noticed that, if we have a \( \text{PH} \) machine \( M \) that computes (say) the \text{PARITY} of a \( 2^n \)-bit oracle string, then by simply reinterpreting the existential quantifiers of \( M \) as OR gates and the universal quantifiers as AND gates, we obtain an \( \text{AC}^0 \) circuit of size \( \text{poly}(n) \) solving the same problem. It follows that, if we can prove a \( 2^\omega(\text{polylog} n) \) lower bound on the size of \( \text{AC}^0 \) circuits computing \text{PARITY}, we can construct an oracle \( A \) relative to which \( \oplus \text{P}^A \not\subseteq \text{PH}^A \). The idea is the same for constructing an \( A \) relative to which \( \mathcal{C}^A \not\subseteq \text{PH}^A \), where \( \mathcal{C} \) is any complexity class.

Indeed, the relation between \( \text{PH} \) and \( \text{AC}^0 \) is so direct that we get the following as a more-or-less immediate counterpart to Theorem 1:

**Theorem 2** *In the unrelativized world (with no oracle), there exists a relation problem solvable in \( \text{BQLOGTIME} \) but not in nonuniform \( \text{AC}^0 \).*

Here \( \text{BQLOGTIME} \) is the class of problems solvable by logarithmic-size quantum circuits, with random access to the input string \( x_1 \ldots x_n \). It should not be confused with the much larger class \( \text{BQNC} \), of problems solvable by logarithmic-depth quantum circuits.

The relation problem that we use to separate \( \text{BQP} \) from \( \text{PH} \), and \( \text{BQLOGTIME} \) from \( \text{AC}^0 \), is called \text{FOURIER FISHING}. The problem can be informally stated as follows. We are given oracle access to \( n \) Boolean functions \( f_1, \ldots, f_n : \{0,1\}^n \rightarrow \{-1,1\} \), which we think of as chosen uniformly at random. The task is to output \( n \) strings, \( z_1, \ldots, z_n \in \{0,1\}^n \), such that the corresponding squared Fourier coefficients \( \hat{f}_1(z_1)^2, \ldots, \hat{f}_n(z_n)^2 \) are “often much larger than average.” Notice that if \( f_i \) is a random Boolean function, then each of its Fourier coefficients \( \hat{f}_i(z) \) follows a normal distribution—meaning that with overwhelming probability, a constant fraction of the Fourier coefficients will be a constant factor larger than the mean. Furthermore, it is straightforward to create a quantum algorithm that samples each \( z \) with probability proportional to \( \hat{f}_i(z)^2 \), so that larger Fourier coefficients are more likely to be sampled than smaller ones.

On the other hand, computing any specific \( \hat{f}_i(z) \) is easily seen to be equivalent to summing \( 2^n \) bits. By well-known lower bounds on the size of \( \text{AC}^0 \) circuits computing the \text{MAJORITY} function (see Håstad [40] for example), it follows that, for any fixed \( z \), computing \( \hat{f}_i(z) \) cannot be in \( \text{PH} \) as an oracle problem. Unfortunately, this does not directly imply any separation between \( \text{BQP} \) and \( \text{PH} \), since the quantum algorithm does not compute \( \hat{f}_i(z) \) either: it just samples a \( z \) with probability proportional to \( \hat{f}_i(z)^2 \). However, we will show that, if there exists a \( \text{BPP}^{\text{PH}} \) machine \( M \) that even *approximately* simulates the behavior of the quantum algorithm, then one can solve \text{MAJORITY} by means of a nondeterministic reduction—which uses approximate counting to estimate \( \Pr[M \text{ outputs } z] \), and adds a constant number of layers to the \( \text{AC}^0 \) circuit. The central difficulty is that, if \( M \) knew the specific \( z \) for which we were interested in estimating \( \hat{f}_i(z) \), then it could choose adversarially never to output that \( z \). To solve this, we will show that we can “smuggle” a \text{MAJORITY} instance into the estimation of a random Fourier coefficient \( \hat{f}_i(z) \), in such a way that it is information-theoretically impossible for \( M \) to determine which \( z \) we care about.

Our second contribution is to define and study a new black-box decision problem, called \text{FOURIER CHECKING}. Informally, in this problem we are given oracle access to two Boolean functions \( f, g : \{0,1\}^n \rightarrow \{-1,1\} \), and are promised that either

(i) \( f \) and \( g \) are both uniformly random, or
(ii) $f$ is uniformly random, while $g$ is extremely well correlated with $f$’s Fourier transform over $\mathbb{Z}_2^n$ (which we call “forrelated”).

The problem is to decide whether (i) or (ii) is the case.

It is not hard to show that **FOURIER CHECKING** is in **BQP**: basically, one can prepare a uniform superposition over all $x \in \{0,1\}^n$, then query $f$, apply a quantum Fourier transform, query $g$, and check whether one has recovered something close to the uniform superposition. On the other hand, being forrelated seems like an extremely “global” property of $f$ and $g$: one that would not be apparent from querying any small number of $f(x)$ and $g(y)$ values. And thus, one might conjecture that **FOURIER CHECKING** (as an oracle problem) is not in **PH**.

In this paper, we adduce strong evidence for that conjecture. Specifically, we show that for every $k \leq 2^{n/4}$, the forrelated distribution over pairs is $O(k^2/2^{n/2})$-almost $k$-wise independent. By this we mean that, if one had 1/2 prior probability that $f$ and $g$ were uniformly random, and 1/2 prior probability that $f$ and $g$ were forrelated, then even conditioned on any $k$ values of $f$ and $g$, the posterior probability that $f$ and $g$ were forrelated would still be

$$\frac{1}{2} \pm O\left(\frac{k^2}{2^{n/2}}\right).$$

We conjecture that this almost $k$-wise independence property is enough, by itself, to imply that an oracle problem is not in **PH**. We call this the **Generalized Linial-Nisan Conjecture**.

Without the $\pm O(k^2/2^{n/2})$ error term, our conjecture would be equivalent to a famous conjecture in circuit complexity made by Linial and Nisan [31] in 1990. Their conjecture stated that **polylogarithmic independence fools** $\mathbf{AC}^0$: in other words, every probability distribution over $N$-bit strings that is uniform on every small subset of bits, is indistinguishable from the truly uniform distribution by $\mathbf{AC}^0$ circuits. When we began investigating this topic a year ago, even the original Linial-Nisan Conjecture was still open. Since then, Braverman [15] (building on earlier work by Bazzi [8] and Razborov [33]) has given a beautiful proof of that conjecture. In other words, to construct an oracle relative to which $\mathbf{BQP} \not\subseteq \mathbf{PH}$, it now suffices to generalize Braverman’s Theorem from $k$-wise independent distributions to almost $k$-wise independent ones. We believe that this is by far the most promising approach to the **BQP** versus **PH** problem.

Alas, generalizing Braverman’s proof is much harder than one might have hoped. To prove the original Linial-Nisan Conjecture, Braverman showed that every $\mathbf{AC}^0$ function $f : \{0,1\}^n \rightarrow \{0,1\}$ can be well-approximated, in the $L_1$-norm, by **low-degree sandwiching polynomials**: real polynomials $p_l, p_u : \mathbb{R}^n \rightarrow \mathbb{R}$, of degree $O(\log n)$, such that $p_l(x) \leq f(x) \leq p_u(x)$ for all $x \in \{0,1\}^n$. Since $p_l$ and $p_u$ trivially have the same expectation on any $k$-wise independent distribution that they have on the uniform distribution, one can show that $f$ must have almost the same expectation as well. To generalize Braverman’s result from $k$-wise independence to almost $k$-wise independence, we will show that it suffices to construct low-degree sandwiching polynomials that satisfy a certain additional condition. This new condition (which we call “low-fat”) basically says that $p_l$ and $p_u$ must be representable as linear combinations of terms (that is, products of $x_i$’s and $(1-x_i)$’s), in such a way that the sum of the absolute values of the coefficients is bounded—thereby preventing “massive cancellations” between positive and negative terms. Unfortunately, while we know two techniques for approximating $\mathbf{AC}^0$ functions by low-degree polynomials—that of Linial-Mansour-Nisan [30] and that of Razborov [32] and Smolensky [39]—neither technique provides anything like

---

7 Up to unimportant variations in the parameters
the control over coefficients that we need. To construct low-fat sandwiching polynomials, it seems necessary to reprove the LMN and Razborov-Smolensky theorems in a more “conservative,” less “profligate” way. And such an advance seems likely to lead to breakthroughs in circuit complexity and computational learning theory having nothing to do with quantum computing.

Let us mention three further applications of FOURIER CHECKING:

(1) If the Generalized Linial-Nisan Conjecture holds, then just like with FOURIER FISHING, we can “scale down by an exponential,” to obtain a promise problem that is in $\text{BQLOGTIME}$ but not in $\text{AC}^0$.

(2) Without any assumptions, we can prove the new results that there exist oracles relative to which $\text{BQP} \not\subset \text{BPP}_{\text{path}}$ and $\text{BQP} \not\subset \text{SZK}$. We can also reprove all previous oracle separations between $\text{BQP}$ and classical complexity classes in a unified fashion.

(3) Independent of its applications to complexity classes, FOURIER CHECKING yields the largest gap between classical and quantum query complexity yet known. Its bounded-error randomized query complexity is $\Omega(N^{1/4})$ (and we conjecture $\Theta(\sqrt{N})$), while its quantum query complexity is only $O(1)$. Prior to this work, the best known gaps were of the form $N^{\Omega(1)}$ versus $O(\log N)$ (for Simon’s and Shor’s algorithms), or $\Theta(\log N)$ versus $O(1)$ (for the Bernstein-Vazirani problem).\footnote{De Beaudrap, Cleve, and Watrous [10] gave an $N^{\Omega(1)}$ versus $O(1)$ separation, but for a non-standard model of quantum query complexity where one gets to map $|x\rangle$ to $|\sigma(x)\rangle$ for a permutation $\sigma$ (rather than $|x\rangle |y\rangle$ to $|x\rangle |y \oplus \sigma(x)\rangle$ as in the standard model).} Buhrman et al. [17] explicitly raised the open problem of finding a larger gap. We also get a property (namely, the property of being forrelated) that takes $N^{\Omega(1)}$ queries to test classically but only $O(1)$ queries to test quantumly.

To summarize our main conclusions:

**Theorem 3** Assuming the Generalized Linial-Nisan Conjecture, there exists an oracle $A$ relative to which $\text{BQP}^A \not\subset \text{PH}^A$, and there also exists a promise problem in $\text{BQLOGTIME} \setminus \text{AC}^0$. Unconditionally, there exists an oracle $A$ relative to which $\text{BQP}^A \not\subset \text{BPP}_{\text{path}}^A$ and $\text{BQP}^A \not\subset \text{SZK}^A$.

As a candidate problem, FOURIER CHECKING has at least five advantages over the RECURSIVE FOURIER SAMPLING problem of Bernstein and Vazirani [13]. First, it is much simpler to define and reason about. Second, FOURIER CHECKING has the almost $k$-wise independence property, which is not shared by RECURSIVE FOURIER SAMPLING, and which immediately connects the former to general questions about pseudorandomness against constant-depth circuits. Third, FOURIER CHECKING can yield exponential separations between quantum and classical models, rather than just quasipolynomial ones. Fourth, one can hope to use FOURIER CHECKING to give an oracle relative to which $\text{BQP}$ is not in $\text{PH}[n^c]$ (or $\text{PH}$ with $n^c$ alternations) for any fixed $c$; by contrast, RECURSIVE FOURIER SAMPLING is in $\text{PH}[\log n]$. Finally, it is at least conceivable that the quantum algorithm for FOURIER CHECKING is good for something. We leave the challenge of finding an explicit computational problem that “instantiates” FOURIER CHECKING, in the same way that FACTORING and DISCRETE LOGARITHM instantiated Shor’s period-finding problem.
1.3 In Defense of Oracles

This paper is concerned with finding oracles relative to which $\text{BQP}$ outperforms classical complexity classes. As such, it is open to the usual objections: “But don’t oracle results mislead us about the ‘real’ world? What about non-relativizing results like $\text{IP} = \text{PSPACE}$ [35]?”

In our view, it is most helpful to think of oracle separations, not as strange metamathematical claims, but as lower bounds in a concrete computational model that is natural and well-motivated in its own right. The model in question is query complexity, where the resource to be minimized is the number of accesses to a very long input string. When someone gives an oracle $A$ relative to which $\mathcal{C}^A \not\subseteq \mathcal{D}^A$, what they really mean is simply that they have found a problem that $\mathcal{C}$ machines can solve using superpolynomially fewer queries than $\mathcal{D}$ machines. In other words, $\mathcal{C}$ has “cleared the first possible obstacle”—the query complexity obstacle—to having capabilities beyond those of $\mathcal{D}$. Of course, it could be (and sometimes is) that $\mathcal{C} \subseteq \mathcal{D}$ for other reasons, but if we do not even have a query complexity lower bound, then proving one is in some sense the obvious place to start.

Oracle separations have played a role in many of the central developments of both classical and quantum complexity theory. As mentioned earlier, proving query complexity lower bounds for $\text{PH}$ machines is essentially equivalent to proving size lower bounds for $\text{AC}^0$ circuits—and indeed, the pioneering $\text{AC}^0$ lower bounds of the early 1980s were explicitly motivated by the goal of proving oracle separations for $\text{PH}$.

Within quantum computing, oracle results have played an even more decisive role: the first evidence for the power of quantum computers came from the oracle separations of Bernstein-Vazirani [13] and Simon [38], and Shor’s algorithm [37] contains an oracle algorithm (for the PERIOD-FINDING problem) at its core.

Having said all that, if for some reason one still feels averse to the language of oracles, then (as mentioned before) one is free to scale everything down by an exponential, and to reinterpret a relativized separation between $\text{BQP}$ and $\text{PH}$ as an unrelativized separation between $\text{BQLOGTIME}$ and $\text{AC}^0$.

2 Preliminaries

It will be convenient to consider Boolean functions of the form $f : \{0,1\}^n \rightarrow \{-1,1\}$. Throughout this paper, we let $N = 2^n$; we will often view the truth table of a Boolean function as an “input” of size $N$. Given a Boolean function $f : \{0,1\}^n \rightarrow \{-1,1\}$, the Fourier transform of $f$ is defined as

$$\hat{f}(z) := \frac{1}{\sqrt{N}} \sum_{x \in \{0,1\}^n} (-1)^{x \cdot z} f(x).$$

Recall Parseval’s identity:

$$\sum_{x \in \{0,1\}^n} f(x)^2 = \sum_{z \in \{0,1\}^n} \hat{f}(z)^2 = N.$$

2.1 Problems

We first define the FOURIER FISHING problem, in both “distributional” and “promise” versions. In the distributional version, we are given oracle access to $n$ Boolean functions $f_1, \ldots, f_n : \{0,1\}^n \rightarrow \{-1,1\}$ and a probability distribution $\mathcal{D}$ over $\{0,1\}^n$. The task is to find an index $i$ such that $f_i$ differs from the majority of $N$ functions.

The promise version of the problem is as follows. We are given oracle access to $n$ Boolean functions $f_1, \ldots, f_n : \{0,1\}^n \rightarrow \{-1,1\}$ and a promise $\epsilon > 0$ such that at least $\epsilon$ of the functions disagree. The task is to find an index $i$ such that $f_i$ differs from at least $\epsilon n$ of the functions.
\{−1, 1\}, which are chosen uniformly and independently at random. The task is to output \( n \) strings, \( z_1, \ldots, z_n \in \{0, 1\}^n \), at least 75\% of which satisfy \(|\hat{f}_i(z_i)| \geq 1\) and at least 25\% of which satisfy \(|\hat{f}_i(z_i)| \geq 2\). (Note that these thresholds are not arbitrary, but were carefully chosen to produce a separation between the quantum and classical models!)

We now want a version of Fourier Fishing that removes the need to assume the \( f_i \)'s are uniformly random, replacing it with a worst-case promise on the \( f_i \)'s. Call an \( n \)-tuple \( \langle f_1, \ldots, f_n \rangle \) of Boolean functions \textit{good} if

\[
\sum_{i=1}^{n} \sum_{z_i: |\hat{f}_i(z_i)| \geq 1} \hat{f}_i(z_i)^2 \geq 0.8Nn,
\]

\[
\sum_{i=1}^{n} \sum_{z_i: |\hat{f}_i(z_i)| \geq 2} \hat{f}_i(z_i)^2 \geq 0.26Nn.
\]

(We will show in Lemma 8 that the vast majority of \( \langle f_1, \ldots, f_n \rangle \) are good.) In Promise Fourier Fishing, we are given oracle access to Boolean functions \( f_1, \ldots, f_n : \{0, 1\}^n \rightarrow \{-1, 1\} \), which are promised to be good. The task, again, is to output strings \( z_1, \ldots, z_n \in \{0, 1\}^n \), at least 75\% of which satisfy \(|\hat{f}_i(z_i)| \geq 1\) and at least 25\% of which satisfy \(|\hat{f}_i(z_i)| \geq 2\).

Next we define a decision problem called Fourier Checking. Here we are given oracle access to two Boolean functions \( f, g : \{0, 1\}^n \rightarrow \{-1, 1\} \). We are promised that either

(i) \( \langle f, g \rangle \) was drawn from the uniform distribution \( U \), which sets every \( f(x) \) and \( g(y) \) by a fair, independent coin toss.

(ii) \( \langle f, g \rangle \) was drawn from the “forrelated” distribution \( F \), which is defined as follows. First choose a random real vector \( v = (v_x)_{x \in \{0, 1\}^n} \in \mathbb{R}^N \), by drawing each entry independently from a Gaussian distribution with mean 0 and variance 1. Then set \( f(x) := \text{sgn}(v_x) \) and \( g(x) := \text{sgn}(\hat{v}_x) \) for all \( x \). Here

\[
\text{sgn}(\alpha) := \begin{cases} 1 & \text{if } \alpha \geq 0 \\ -1 & \text{if } \alpha < 0 \end{cases}
\]

and \( \hat{v} \) is the Fourier transform of \( v \) over \( \mathbb{Z}_2^n \):

\[
\hat{v}_y := \frac{1}{\sqrt{N}} \sum_{x \in \{0, 1\}^n} (-1)^{x \cdot y} v_x.
\]

In other words, \( f \) and \( g \) individually are still uniformly random, but they are no longer independent: now \( g \) is extremely well correlated with the Fourier transform of \( f \) (hence “forrelated”).

The problem is to accept if \( \langle f, g \rangle \) was drawn from \( F \), and to reject if \( \langle f, g \rangle \) was drawn from \( U \). Note that, since \( F \) and \( U \) overlap slightly, we can only hope to succeed with overwhelming probability over the choice of \( \langle f, g \rangle \), not for every \( \langle f, g \rangle \) pair.
We can also define a promise-problem version of Fourier Checking. In Promise Fourier Checking, we are promised that the quantity

\[ p(f, g) := \frac{1}{N^3} \left( \sum_{x,y \in \{0,1\}^n} f(x)(-1)^{x \cdot y} g(y) \right)^2 \]

is either at least 0.05 or at most 0.01. The problem is to accept in the former case and reject in the latter case.

### 2.2 Complexity Classes

See the Complexity Zoo\(^\text{10}\) for the definitions of standard complexity classes, such as \( \text{BQP}, \text{AM}, \) and \( \text{PH} \). When we write \( \mathcal{C}^{\text{PH}} \) (i.e., a complexity class \( \mathcal{C} \) with an oracle for the polynomial hierarchy), we mean \( \cup_{k \geq 1} \mathcal{C}^{\Sigma^P_k} \).

We will consider not only decision problems, but also relation problems (also called function problems). In a relation problem, the output is not a single bit but a poly \((n)\)-bit string \( y \). There could be many valid \( y \)'s for a given instance, and the algorithm’s task is to output any one of them.

The definitions of \( \text{FP} \) and \( \text{FNP} \) (the relation versions of \( \text{P} \) and \( \text{NP} \)) are standard. We now define \( \text{FBPP} \) and \( \text{FBQP} \), the relation versions of \( \text{BPP} \) and \( \text{BQP} \).

**Definition 4** FBPP is the class of relations \( R \subseteq \{0,1\}^* \times \{0,1\}^* \) for which there exists a probabilistic polynomial-time algorithm \( A \) that, given any input \( x \in \{0,1\}^n \), produces an output \( y \) such that

\[ \Pr \left[ (x, y) \in R \right] = 1 - o(1), \]

where the probability is over \( A \)'s internal randomness. (In particular, this implies that for every \( x \), there exists at least one \( y \) such that \( (x, y) \in R \).) FBQP is defined the same way, except that \( A \) is a quantum algorithm rather than a classical one.

An important point about FBPP and FBQP is that, as far as we know, these classes do not admit amplification. In other words, the value of an algorithm’s success probability might actually matter, not just the fact that the probability is bounded above \( 1/2 \). This is why we adopt the convention that an algorithm “succeeds” if it outputs \( (x, y) \in R \) with probability \( 1 - o(1) \). In practice, we will give oracle problems for which the FBQP algorithm succeeds with probability \( 1 - 1/\exp(n) \), while any FBPP\(^{\text{PH}} \) algorithm succeeds with probability at most (say) 0.99. How far the constant in this separation can be improved is an open problem.

Another important point is that, while \( \text{BPP}^{\text{PH}} = \text{P}^{\text{PH}} \) (which follows from \( \text{BPP} \subseteq \Sigma^P_2 \)), the class \( \text{FBPP}^{\text{PH}} \) is strictly larger than \( \text{FP}^{\text{PH}} \). To see this, consider the relation

\[ R = \{(0^n, y) : K(y) \geq n \}, \]

where we are given \( n \), and asked to output any string of Kolmogorov complexity at least \( n \). Clearly this problem is in FBPP: just output a random \( 2n \)-bit string. On the other hand, just as obviously the problem is not in FP\(^{\text{PH}} \). This is why we need to construct an oracle \( A \) such that \( \text{FBQP}^A \not\subseteq \text{FBPP}^{\text{PH}A} \): because constructing an oracle \( A \) such that \( \text{FBQP}^A \not\subseteq \text{FP}^{\text{PH}A} \) is trivial and not even related to quantum computing.

\(^{10}\)www.complexityzoo.com
We now discuss some “low-level” complexity classes. \( \text{AC}^0 \) is the class of problems solvable by a nonuniform family of AND/OR/NOT circuits, with depth \( O(1) \), size \( \text{poly} (n) \), and unbounded fanin. When we say “\( \text{AC}^0 \) circuit,” we mean a constant-depth circuit of AND/OR/NOT gates, not necessarily of polynomial size. Any such circuit can be made into a formula (i.e., a circuit of fanout 1) with only a polynomial increase in size. The circuit has depth \( d \) if it consists of \( d \) alternating layers of AND and OR gates (without loss of generality, the NOT gates can all be pushed to the bottom, and we do not count them towards the depth). For example, a DNF (Disjunctive Normal Form) formula is just an \( \text{AC}^0 \) circuit of depth 2.

We will also be interested in quantum logarithmic time, which can be defined naturally as follows:

**Definition 5** \( \text{BQLOGTIME} \) is the class of languages \( L \subseteq \{0,1\}^* \) that are decidable, with bounded probability of error, by a \( \text{LOGTIME} \)-uniform family of quantum circuits \( \{C_n\}_n \) such that each \( C_n \) has \( O(\log n) \) gates, and can include gates that make random-access queries to the input string \( x = x_1 \ldots x_n \) (i.e., that map \( |i\rangle |z\rangle \) to \( |i\rangle |z \oplus x_i\rangle \) for every \( i \in [n] \)).

One other complexity class that arises in this paper, which is less well known than it should be, is \( \text{BPP} \) path. Loosely speaking, \( \text{BPP} \) path can be defined as the class of problems that are solvable in probabilistic polynomial time, given the ability to “postselect” (that is, discard all runs of the computation that do not produce a desired result, even if such runs are the overwhelming majority). Formally:

**Definition 6** \( \text{BPP} \) path is the class of languages \( L \subseteq \{0,1\}^* \) for which there exists a \( \text{BPP} \) machine \( M \), which can either “succeed” or “fail” and conditioned on succeeding either “accept” or “reject,” such that for all inputs \( x \):

(i) \( \Pr [M (x) \text{ succeeds}] > 0 \).

(ii) \( x \in L \implies \Pr [M (x) \text{ accepts } | M (x) \text{ succeeds}] \geq \frac{2}{3} \).

(iii) \( x \notin L \implies \Pr [M (x) \text{ accepts } | M (x) \text{ succeeds}] \leq \frac{1}{3} \).

\( \text{BPP} \) path was defined by Han, Hemaspaandra, and Thierauf [28], who also showed that \( \text{MA} \subseteq \text{BPP} \) path and \( \text{P}^{\text{NP}} \subseteq \text{BPP} \) path \( \subseteq \text{BPP}^{\text{NP}} \). Using \text{FOURIER CHECKING}, we will construct an oracle \( A \) relative to which \( \text{BQP}^A \not\subset \text{BPP}^A \) path. This result might not sound amazing, but (i) it is new, (ii) it does not follow from the “standard” quantum algorithms, such as those of Simon [38] and Shor [37], and (iii) it supersedes almost all previous oracle results placing \( \text{BQP} \) outside classical complexity classes. As another illustration of the versatility of \text{FOURIER CHECKING}, we use it to give an \( A \) such that \( \text{BQP}^A \not\subset \text{SZK}^A \), where \( \text{SZK} \) is Statistical Zero Knowledge. The opposite direction—an \( A \) such that \( \text{SZK}^A \not\subset \text{BQP}^A \)—was shown by Aaronson [1] in 2002.

# 3 Quantum Algorithms

In this section, we show that \text{FOURIER FISHING} and \text{FOURIER CHECKING} both admit simple quantum algorithms.

---

\[11\] The one exception is the result of Green and Pruim [27] that there exists an \( A \) relative to which \( \text{BQP}^A \not\subset \text{P}^{\text{NP},A} \), but that can also be easily reproduced using \text{FOURIER CHECKING}.
Figure 1: The Fourier coefficients of a random Boolean function follow a Gaussian distribution, with mean 0 and variance 1. However, larger Fourier coefficients are more likely to be observed by the quantum algorithm.

3.1 Quantum Algorithm for Fourier Fishing

Here is a quantum algorithm, $\text{FF-ALG}$, that solves $\text{FOURIER FISHING}$ with overwhelming probability in $O(n^2)$ time and $n$ quantum queries (one to each $f_i$). For $i := 1$ to $n$, first prepare the state

$$\frac{1}{\sqrt{N}} \sum_{x \in \{0,1\}^n} f_i(x) |x\rangle,$$

then apply Hadamard gates to all $n$ qubits, then measure in the computational basis and output the result as $z_i$.

Intuitively, $\text{FF-ALG}$ samples the Fourier coefficients of each $f_i$ under a distribution that is skewed towards larger coefficients; the algorithm’s behavior is illustrated pictorially in Figure 1. We now give a formal analysis. Recall the definition of a “good” tuple $\langle f_1, \ldots, f_n \rangle$ from Section 2.1. Assuming $\langle f_1, \ldots, f_n \rangle$ is good, it is easy to analyze $\text{FF-ALG}$’s success probability.

**Lemma 7** Assuming $\langle f_1, \ldots, f_n \rangle$ is good, $\text{FF-ALG}$ succeeds with probability $1 - 1/\exp(n)$.

**Proof.** Let $\langle z_1, \ldots, z_n \rangle$ be the algorithm’s output. For each $i$, let $X_i$ be the event that $|\hat{f}_i(z_i)| \geq 1$ and let $Y_i$ be the event that $|\hat{f}_i(z_i)| \geq 2$. Also let $p_i := \Pr[X_i]$ and $q_i := \Pr[Y_i]$, where the probability is over $\text{FF-ALG}$’s internal (quantum) randomness. Then clearly

$$p_i = \frac{1}{N} \sum_{z_i : |\hat{f}_i(z_i)| \geq 1} \hat{f}_i(z_i)^2,$$

$$q_i = \frac{1}{N} \sum_{z_i : |\hat{f}_i(z_i)| \geq 2} \hat{f}_i(z_i)^2.$$
So by assumption,
\[ p_1 + \cdots + p_n \geq 0.8n, \]
\[ q_1 + \cdots + q_n \geq 0.26n. \]

By a Chernoff/Hoeffding bound, it follows that
\[
\Pr \left[ X_1 + \cdots + X_n \geq 0.75n \right] > 1 - \frac{1}{\exp(n)},
\]
\[
\Pr \left[ Y_1 + \cdots + Y_n \geq 0.25n \right] > 1 - \frac{1}{\exp(n)}.
\]

Hence FF-ALG succeeds with \( 1 - 1/\exp(n) \) probability by the union bound.

Similarly, it follows that
\[
\Pr \left[ Z_1 + \cdots + Z_n \geq 0.8n \right] > 1 - \frac{1}{\exp(n)},
\]
\[
\Pr \left[ W_1 + \cdots + W_n \geq 0.26n \right] > 1 - \frac{1}{\exp(n)}.
\]

Hence FF-ALG succeeds with \( 1 - 1/\exp(n) \) probability by the union bound.

We also have the following:

**Lemma 8** \( \langle f_1, \ldots, f_n \rangle \) is good with probability \( 1 - 1/\exp(n) \), if the \( f_i \)'s are chosen uniformly at random.

**Proof.** Choose \( f : \{0,1\}^n \rightarrow \{-1,1\} \) uniformly at random. Then for each \( z \), the Fourier coefficient \( \hat{f}(z) \) follows a normal distribution, with mean 0 and variance 1. So in the limit of large \( N \),
\[
E_f \left[ \sum_{z:|\hat{f}(z)| \geq 1} \hat{f}(z)^2 \right] = \sum_{z \in \{0,1\}^n} \Pr \left[ |\hat{f}(z)| \geq 1 \right] E \left[ \hat{f}(z)^2 \mid |\hat{f}(z)| \geq 1 \right]
\]
\[
\approx \frac{2N}{\sqrt{2\pi}} \int_1^\infty e^{-x^2/2}x^2dx
\]
\[
\approx 0.801N.
\]

Likewise,
\[
E_f \left[ \sum_{z:|\hat{f}(z)| \geq 2} \hat{f}(z)^2 \right] \approx \frac{2N}{\sqrt{2\pi}} \int_2^\infty e^{-x^2/2}x^2dx
\]
\[
\approx 0.261N.
\]

Since the \( f_i \)'s are chosen independently of one another, it follows by a Chernoff bound that
\[
\sum_{i=1}^n \sum_{z_i:|\hat{f}_i(z_i)| \geq 1} \hat{f}_i(z_i)^2 \geq 0.8Nn,
\]
\[
\sum_{i=1}^n \sum_{z_i:|\hat{f}_i(z_i)| \geq 2} \hat{f}_i(z_i)^2 \geq 0.26Nn
\]
with probability \( 1 - 1/\exp(n) \) over the choice of \( \langle f_1, \ldots, f_n \rangle \).

Combining Lemmas 7 and 8, we find that FF-ALG succeeds with probability \( 1 - 1/\exp(n) \), where the probability is over both \( \langle f_1, \ldots, f_n \rangle \) and FF-ALG’s internal randomness.
3.2 Quantum Algorithm for Fourier Checking

We now turn to Fourier Checking, the problem of deciding whether two Boolean functions \( f, g \) are independent or forrelated. Here is a quantum algorithm, \( \text{FC-ALG} \), that solves Fourier Checking with constant error probability using \( O(1) \) queries. First prepare a uniform superposition over all \( x \in \{0, 1\}^n \). Then query \( f \) in superposition, to create the state

\[
\frac{1}{\sqrt{N}} \sum_{x \in \{0, 1\}^n} f(x) |x\rangle
\]

Then apply Hadamard gates to all \( n \) qubits, to create the state

\[
\frac{1}{N} \sum_{x, y \in \{0, 1\}^n} f(x) (-1)^{x \cdot y} |y\rangle.
\]

Then query \( g \) in superposition, to create the state

\[
\frac{1}{N} \sum_{x, y \in \{0, 1\}^n} f(x) (-1)^{x \cdot y} g(y) |y\rangle.
\]

Then apply Hadamard gates to all \( n \) qubits again, to create the state

\[
\frac{1}{N^{3/2}} \sum_{x, y, z \in \{0, 1\}^n} f(x) (-1)^{x \cdot y} g(y) (-1)^{y \cdot z} |z\rangle.
\]

Finally, measure in the computational basis, and “accept” if and only if the outcome \( |0\rangle^\otimes n \) is observed. If needed, repeat the whole algorithm \( O(1) \) times to boost the success probability.

It is clear that the probability of observing \( |0\rangle^\otimes n \) (in a single run of FC-ALG) equals

\[
p(f, g) := \frac{1}{N^3} \left( \sum_{x, y \in \{0, 1\}^n} f(x) (-1)^{x \cdot y} g(y) \right)^2.
\]

Recall that Promise Fourier Checking was the problem of deciding whether \( p(f, g) \geq 0.05 \) or \( p(f, g) \leq 0.01 \), promised that one of these is the case. Thus, we immediately get a quantum algorithm to solve Promise Fourier Checking, with constant error probability, using \( O(1) \) queries to \( f \) and \( g \).

For the distributional version of Fourier Checking, we also need the following theorem.

**Theorem 9** If \( \langle f, g \rangle \) is drawn from the uniform distribution \( \mathcal{U} \), then

\[
\mathbb{E}_{\mathcal{U}}[p(f, g)] = \frac{1}{N}.
\]

If \( \langle f, g \rangle \) is drawn from the forrelated distribution \( \mathcal{F} \), then

\[
\mathbb{E}_{\mathcal{F}}[p(f, g)] > 0.07.
\]
Proof. The first part follows immediately by symmetry (i.e., the fact that all \( N = 2^n \) measurement outcomes of the quantum algorithm are equally likely).

For the second part, let \( v \in \mathbb{R}^N \) be the vector of independent Gaussians used to generate \( f \) and \( g \), let \( w = v / \|v\|_2 \) be \( v \) scaled to have unit norm, and let \( H \) be the \( n \)-qubit Hadamard matrix. Also let \( \text{flat} (w) \) be the unit vector whose \( x^{\text{th}} \) entry is \( \text{sgn} (w_x) / \sqrt{N} = f (x) / \sqrt{N} \), and let \( \text{flat} (Hw) \) be the unit vector whose \( x^{\text{th}} \) entry is \( \text{sgn} (\tilde{v}_x) / \sqrt{N} = g (x) / \sqrt{N} \). Then \( p (f, g) \) equals

\[
\left( \text{flat} (w)^T H \text{flat} (Hw) \right)^2,
\]
or the squared inner product between the vectors \( \text{flat} (w) \) and \( H \text{flat} (Hw) \). Note that \( w^T \cdot HHw = w^T w = 1 \). So the whole problem is to understand the “discretization error” incurred in replacing \( w^T \) by \( \text{flat} (w)^T \) and \( HHw \) by \( H \text{flat} (Hw) \). By the triangle inequality, the angle between \( \text{flat} (w) \) and \( H \text{flat} (Hw) \) is at most the angle between \( \text{flat} (w) \) and \( w \), plus the angle between \( w \) and \( H \text{flat} (Hw) \). In other words:

\[
\arccos \left( \text{flat} (w)^T H \text{flat} (Hw) \right) \leq \arccos \left( \text{flat} (w)^T w \right) + \arccos \left( w^T H \text{flat} (Hw) \right).
\]

Now,

\[
\text{flat} (w)^T w = \sum_{x \in \{0,1\}^n} w_x \cdot \frac{|w_x|}{\sqrt{N}} w_x = \frac{1}{\sqrt{N}} \sum_{x \in \{0,1\}^n} |w_x| = \frac{\sum_{x \in \{0,1\}^n} |v_x|}{\sqrt{N} \|v\|_2}.
\]

Recall that each \( v_x \) is an independent real Gaussian with mean 0 and variance 1, meaning that each \( |v_x| \) is an independent nonnegative random variable with expectation \( \sqrt{2/\pi} \). So by standard tail bounds, for all constants \( \varepsilon > 0 \) we have

\[
\Pr_v \left[ \sum_{x \in \{0,1\}^n} |v_x| \leq \left( \sqrt{\frac{2}{\pi}} - \varepsilon \right) N \right] \leq \frac{1}{\exp (N)},
\]

\[
\Pr \left[ \|v\|_2^2 \geq (1 + \varepsilon) N \right] \leq \frac{1}{\exp (N)}.
\]

So by the union bound,

\[
\Pr_v \left[ \text{flat} (w)^T w \leq \sqrt{\frac{2}{\pi}} - \varepsilon \right] \leq \frac{1}{\exp (N)}.
\]

Since \( H \) is unitary, the same analysis applies to \( w^T H \text{flat} (Hw) \). Therefore, for all constants \( \varepsilon > 0 \), with \( 1 - 1/\exp (N) \) probability we have

\[
\arccos \left( \text{flat} (w)^T w \right) \leq \left( \arccos \sqrt{\frac{2}{\pi}} \right) + \varepsilon,
\]

\[
\arccos \left( w^T H \text{flat} (Hw) \right) \leq \left( \arccos \sqrt{\frac{2}{\pi}} \right) + \varepsilon.
\]
So setting $\varepsilon = 0.0001$, 
\[
\arccos \left( \text{flat}(w)^T H \text{flat}(Hw) \right) \leq \arccos \left( \text{flat}(w)^T w \right) + \arccos \left( w^T H \text{flat}(Hw) \right) \\
\leq 2 \left( \arccos \sqrt{\frac{2}{\pi}} \right) + 2\varepsilon \\
\leq 1.3
\]

Therefore, with $1 - 1/\exp(N)$ probability over $\langle f, g \rangle$ drawn from $\mathcal{F}$,
\[
\text{flat}(w)^T H \text{flat}(Hw) \geq \cos 1.3,
\]
in which case $p(f, g) \geq (\cos 1.3)^2 \approx 0.072$. □

Combining Theorem 9 with Markov’s inequality, we immediately get the following:

**Corollary 10**

\[
\Pr_{\langle f,g \rangle \sim U} [p(f, g) \geq 0.01] \leq \frac{100}{N}, \\
\Pr_{\langle f,g \rangle \sim D} [p(f, g) \geq 0.05] \geq \frac{1}{50}.
\]

## 4 The Classical Complexity of Fourier Fishing

In Section 3.1, we gave a quantum algorithm for **Fourier Fishing** that made only one query to each $f_i$. By contrast, it is not hard to show that any classical algorithm for **Fourier Fishing** requires exponentially many queries to the $f_i$’s—or in complexity terms, that **Fourier Fishing** is not in $\text{FBPP}$. In this section, we prove a much stronger result: that **Fourier Fishing** is not even in $\text{FBPP}^{\text{PH}}$. This result does not rely on any unproved conjectures.

### 4.1 Constant-Depth Circuit Lower Bounds

Our starting point will be the following $\text{AC}^0$ lower bound, which can be found in the book of Håstad [40] for example.

**Theorem 11 ([40])** Any depth-$d$ circuit that accepts all $n$-bit strings of Hamming weight $n/2 + 1$, and rejects all strings of Hamming weight $n/2$, has size $\exp(\Omega(1/(d-1)))$.

We now give a corollary of Theorem 11, which (though simple) seems to be new, and might be of independent interest. Consider the following problem, which we call $\varepsilon$-Bias Detection. We are given a string $y = y_1 \ldots y_m \in \{0, 1\}^m$, and are promised that each bit $y_i$ is 1 with independent probability $p$. The task is to decide whether $p = 1/2$ or $p = 1/2 + \varepsilon$.

**Corollary 12** Let $\mathcal{U}[\varepsilon]$ be the distribution over $\{0, 1\}^m$ where each bit is 1 with independent probability $1/2 + \varepsilon$. Then any depth-$d$ circuit $C$ such that
\[
\Pr_{x \sim \mathcal{U}[\varepsilon]} [C(x)] - \Pr_{x \sim \mathcal{U}[0]} [C(x)] = \Omega(1)
\]
has size $\exp(\Omega(1/(d+2)))$. 

16
Proof. Suppose such a distinguishing circuit $C$ exists, with depth $d$ and size $S$, for some $\varepsilon > 0$ (the parameter $m$ is actually irrelevant). Let $n = 1/\varepsilon$, and assume for simplicity that $n$ is an integer. Using $C$, we will construct a new circuit $C'$ with depth $d' = d + 3$ and size $S' = O(nS) + \text{poly}(n)$, which accepts all strings $x \in \{0, 1\}^n$ of Hamming weight $n/2 + 1$, and rejects all strings of Hamming weight $n/2$. By Theorem 11, this will imply that the original circuit $C$ must have had size

$$S = \frac{1}{n} \exp \left( \Omega \left( n^{1/(d'-1)} \right) \right) - \text{poly}(n)$$

$$= \exp \left( \Omega \left( \frac{1}{\varepsilon^{1/(d+2)}} \right) \right).$$

So fix an input $x \in \{0, 1\}^n$, and suppose we choose $m$ bits $x_{i_1}, \ldots, x_{i_m}$ from $x$, with each index $i_j$ chosen uniformly at random with replacement. Call the resulting $m$-bit string $y$. Observe that if $x$ had Hamming weight $n/2$, then $y$ will be distributed according to $U[0]$, while if $x$ had Hamming weight $n/2 + 1$, then $y$ will be distributed according to $U[\varepsilon]$. So by assumption,

$$\Pr[C(y) \mid |x| = n/2] = \alpha,$$

$$\Pr[C(y) \mid |x| = n/2 + 1] = \alpha + \delta$$

for some constants $\alpha$ and $\delta \neq 0$ (we can assume $\delta > 0$ without loss of generality).

Now suppose we repeat the above experiment $T = kn$ times, for some constant $k = k(\alpha, \delta)$. That is, we create $T$ strings $y_1, \ldots, y_T$ by choosing random bits of $x$, so that each $y_i$ is distributed independently according to $U[0]$ if $|x| = n/2$ or $U[\varepsilon]$ if $|x| = n/2 + 1$. We then apply $C$ to each $y_i$. Let

$$Z = C(y_1) + \cdots + C(y_T)$$

be the number of $C$ invocations that accept. Then by a Chernoff bound, if $|x| = n/2$ then

$$\Pr \left[ Z > \alpha T + \frac{\delta}{3} T \right] < \exp(-n),$$

while if $|x| = n/2 + 1$ then

$$\Pr \left[ Z < \alpha T + \frac{2\delta}{3} T \right] < \exp(-n).$$

By taking $k$ large enough, we can make both of these probabilities less than $2^{-n}$. By the union bound, this implies that there must exist a way to choose $y_1, \ldots, y_T$ so that

$$|x| = \frac{n}{2} \implies Z \leq \alpha T + \frac{\delta}{3} T,$$

$$|x| = \frac{n}{2} + 1 \implies Z \geq \alpha T + \frac{2\delta}{3} T$$

for every $x$ with $|x| \in \{n/2, n/2 + 1\}$ simultaneously. In forming the circuit $C'$, we simply hardwire that choice.

The last step is to decide whether $Z \leq \alpha T + \frac{\delta}{3} T$ or $Z \geq \alpha T + \frac{2\delta}{3} T$. This can be done using an AC$^0$ circuit for the Approximate Majority problem (see Viola [41] for example), which has depth 3 and size $\text{poly}(T)$. The end result is a circuit $C'$ to distinguish $|x| = n/2$ from $|x| = n/2 + 1$, which has depth $d + 3$ and size $TS + \text{poly}(T) = O(nS) + \text{poly}(n)$. $\blacksquare$
4.2 Secretly Biased Fourier Coefficients

In this section, we prove two lemmas indicating that one can slightly bias one of the Fourier coefficients of a random Boolean function $f : \{0,1\}^n \rightarrow \{-1,1\}$, and yet still have $f$ be indistinguishable from a random Boolean function (so that, in particular, an adversary has no way of knowing which Fourier coefficient was biased). These lemmas will play a key role in our reduction from $\varepsilon$-Bias Detection to Fourier Fishing.

Fix a string $s \in \{0,1\}^n$. Let $A[s]$ be the probability distribution over functions $f : \{0,1\}^n \rightarrow \{-1,1\}$ where each $f(x)$ is 1 with independent probability $\frac{1}{2} + (-1)^{s \cdot x} \frac{1}{2\sqrt{N}}$, and let $B[s]$ be the distribution where each $f(x)$ is 1 with independent probability $\frac{1}{2} - (-1)^{s \cdot x} \frac{1}{2\sqrt{N}}$. Then let $D[s] = \frac{1}{2} (A[s] + B[s])$ (that is, an equal mixture of $A[s]$ and $B[s]$).

Now consider the following game involving Alice and Bob, which closely models our problem. First, Alice chooses a string $s \in \{0,1\}^n$ uniformly at random. She then draws $f$ according to $D[s]$. She keeps $s$ secret, and sends the truth table of $f$ to Bob. After examining the entire truth table, Bob can output any string $z \in \{0,1\}^n$ such that $|\hat{f}(z)| \geq \beta$. Bob’s goal is to avoid outputting $s$. The following lemma says that, regardless of Bob’s strategy, if $\beta$ is sufficiently greater than 1, then Bob must output $s$ with probability significantly greater than $1/N$.

**Lemma 13** In the above game, we have

$$\Pr[s = z] \geq \frac{e^\beta + e^{-\beta}}{2\sqrt{eN}}.$$ 

where the probability is over $s$, $f$, and Bob’s random choices.

The intuition behind Lemma 13 is simple: $f$ looks to Bob almost exactly like a random Boolean function. Let $V_\beta$ be the set of all $s \in \{0,1\}^n$ such that $|\hat{f}(s)| \geq \beta$. Then when Alice biases $f$, of course she significantly increases the probability that $s \in V_\beta$. But the biased Fourier coefficient $\hat{f}(s)$ still gets “lost in the noise”—that is, it is information-theoretically impossible for Bob to tell $s$ apart from the exponentially many other strings $z$ that are in $V_\beta$ just by chance. Therefore, the best Bob can do is basically just to output a random element of $V_\beta$. But since $|V_\beta| \ll N$, this strategy will cause Bob to output $s$ with probability $\gg 1/N$.

Formalizing the above intuition involves a surprisingly simple Bayesian calculation. Basically, all we need to do is fix $f$ together with Bob’s output $z$ (which we know belongs to $V_\beta$), and then calculate how likely $f$ was to be drawn from $D[z]$ rather than $D[z']$ for some $z' \neq z$.

To see the connection to our problem, think of Bob as a putative $\text{AC}^0$ circuit for Fourier Fishing, and Alice as an $\text{AC}^0$ reduction that uses Bob to solve $\varepsilon$-Bias Detection. Bob might try not to cooperate, but by choosing $s$ randomly, Alice can force even an adversarial Bob to give her useful information about the magnitude of $\hat{f}(s)$.

**Proof of Lemma 13.** By Yao’s principle, we can assume without loss of generality that Bob’s strategy is deterministic. For each $z$, let $F[z]$ be the set of all $f$’s that cause Bob to output $z$. Then the first step is to lower-bound $\Pr_{D[z]}[f]$, for some fixed $z$ and $f \in F[z]$. Let $N_f[z]$ be the number of inputs $x \in \{0,1\}^n$ such that $f(x) = (-1)^{z \cdot x}$. It is not hard to see that $N_f[z] = \frac{N}{2} + \frac{\sqrt{N} f(z)}{2}$. 

18
So
\[
\Pr_{D[z]}[f] = \frac{1}{2} \left( \Pr_{A[z]}[f] + \Pr_{B[z]}[f] \right) \\
= \frac{1}{2} \left( \prod_{x \in \{0,1\}^n} \left( \frac{1}{2} + \frac{(-1)^{z \cdot x} f(x)}{2\sqrt{N}} \right) + \prod_{x \in \{0,1\}^n} \left( \frac{1}{2} - \frac{(-1)^{z \cdot x} f(x)}{2\sqrt{N}} \right) \right) \\
= \frac{1}{2^{N+1}} \left( \left( 1 + \frac{1}{\sqrt{N}} \right)^{N_{f[z]}} \left( 1 - \frac{1}{\sqrt{N}} \right)^{N - N_{f[z]}} + \left( 1 - \frac{1}{\sqrt{N}} \right)^{N_{f[z]}} \left( 1 + \frac{1}{\sqrt{N}} \right)^{N - N_{f[z]}} \right) \\
= \frac{1}{2^{N+1}} \left( 1 - \frac{1}{N} \right)^{N/2} \left( \left( 1 + \frac{1}{\sqrt{N}} \right)^{\sqrt{Nf(z)/2}} + \left( 1 - \frac{1}{\sqrt{N}} \right)^{\sqrt{Nf(z)/2}} \right) \\
\geq \frac{1}{2\sqrt{e^2N}} \left( \hat{f}(z) + e^{-\hat{f}(z)} \right) \\
\geq e^\beta + e^{-\beta}.
\]

Here the last line follows from the assumption \( |\hat{f}(z)| \geq \beta \), together with the fact that \( e^y + e^{-y} \) increases monotonically away from \( y = 0 \).

Summing over all \( z \) and \( f \),
\[
\Pr[s = z] = \sum_{z \in \{0,1\}^n} \sum_{f \in \mathcal{F}[z]} \Pr[f] \cdot \Pr[s = z \mid f] \\
= \sum_{z \in \{0,1\}^n} \sum_{f \in \mathcal{F}[z]} \Pr[f] \cdot \Pr[f \mid s = z] \Pr[s = z \mid f] / \Pr[f] \\
= \frac{1}{N} \sum_{z \in \{0,1\}^n} \sum_{f \in \mathcal{F}[z]} \Pr_{D[z]}[f] \\
\geq \frac{e^\beta + e^{-\beta}}{2\sqrt{e^2N}}.
\]

Now let \( D = E_s[D[s]] \) (that is, an equal mixture of all the \( D[s] \)'s). We claim that \( D \) is extremely close in variation distance to \( U \), the uniform distribution over all Boolean functions \( f : \{0,1\}^n \rightarrow \{-1,1\} \).

**Lemma 14** \( \|D - U\| = O \left( \frac{1}{\sqrt{N}} \right) \).

**Proof.** By a calculation from Lemma 13, for all \( f \) and \( s \) we have
\[
\Pr_{D[s]}[f] = \frac{1}{2\sqrt{e^2N}} \left( \hat{f}(s) + e^{-\hat{f}(s)} \right)
\]
in the limit of large \( N \). Hence
\[
\Pr_D[f] = E_s \left[ \Pr_{D[s]}[f] \right] = \frac{1}{2\sqrt{e^2N^2}} \sum_{s \in \{0,1\}^n} \left( \hat{f}(s) + e^{-\hat{f}(s)} \right).
\]

19
Clearly \(\mathbb{E}_f [\Pr_D [f]] = 1/2^N\). Our goal is to upper-bound the variance \(\text{Var}_f [\Pr_D [f]]\), which measures the distance from \(D\) to the uniform distribution. In the limit of large \(N\), we have

\[
\mathbb{E}_f \left[ \Pr_D [f]^2 \right] = \frac{1}{4eN^22^{2N}} \left( \sum_s \mathbb{E}_f \left[ (e^{f(s)} + e^{-\hat{f}(s)})^2 \right] + \sum_{s \neq t} \mathbb{E}_f \left[ (e^{f(s)} + e^{-\hat{f}(s)}) (e^{f(t)} + e^{-\hat{f}(t)}) \right] \right)
\]

\[
= \frac{1}{4eN^22^{2N}} \left( \sum_s \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} e^{-x^2/2} (e^x + e^{-x})^2 \, dx + \sum_{s \neq t} \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} e^{-x^2/2} (e^x + e^{-x}) \, dx \right)
\]

\[
= \frac{1}{4eN^22^{2N}} \left( (2e^2 + 2) N + 4eN (N - 1) \right) = \frac{1}{2^{2N}} \left( 1 + \frac{(e - 1)^2}{2eN} \right).
\]

Hence

\[
\text{Var}_f \left[ \Pr_D [f] \right] = \mathbb{E}_f \left[ \Pr_D [f]^2 \right] - \mathbb{E}_f \left[ \Pr_D [f] \right]^2 = \frac{(e - 1)^2}{2eN2^{2N}}.
\]

So by Cauchy-Schwarz,

\[
\mathbb{E}_f \left[ \left| \Pr_D [f] - \Pr_U [f] \right| \right] \leq \sqrt{\text{Var}_f \left[ \Pr_D [f] \right]} = \frac{e - 1}{\sqrt{2eN}} \cdot \frac{1}{2^N}
\]

and

\[
\|D - U\| = \frac{1}{2} \sum_f \left| \Pr_D [f] - \Pr_U [f] \right| = O \left( \frac{1}{\sqrt{N}} \right).
\]

An immediate corollary of Lemma 14 is that, if a \textsc{Fourier Fishing} algorithm succeeds with probability \(p\) on \(\langle f_1, \ldots, f_n \rangle\) drawn from \(U^n\), then it also succeeds with probability at least

\[
p - \|D^n - U^n\| \geq p - O \left( \frac{n}{\sqrt{N}} \right)
\]

on \(\langle f_1, \ldots, f_n \rangle\) drawn from \(D^n\).

### 4.3 Putting It All Together

Using the results of Sections 4.1 and 4.2, we are now ready to prove a lower bound on the constant-depth circuit complexity of \textsc{Fourier Fishing}.

**Theorem 15** Any depth-\(d\) circuit that solves the \textsc{Fourier Fishing} problem, with probability at least 0.99 over \(f_1, \ldots, f_n\) chosen uniformly at random, has size \(\exp \left( \Omega \left( N^{1/(2d+8)} \right) \right)\).

**Proof.** Let \(C\) be a circuit of depth \(d\) and size \(s\). Let \(G\) be the set of all \(\langle f_1, \ldots, f_n \rangle\) on which \(C\) succeeds: that is, for which it outputs \(z_1, \ldots, z_n\), at least 75\% of which satisfy \(\hat{f}_i (z_i) \geq 1\) and at least 25\% of which satisfy \(\hat{f}_i (z_i) \geq 2\). Suppose

\[
\Pr_{U^n} [\langle f_1, \ldots, f_n \rangle \in G] \geq 0.99.
\]
Then by Lemma 14, we also have

\[
\Pr_{D^n}[\langle f_1, \ldots, f_n \rangle \in G] \geq 0.99 - O \left( \frac{n}{\sqrt{N}} \right) \geq 0.98
\]

for sufficiently large \( n \).

Using the above fact, we will convert \( C \) into a new circuit \( C' \) that solves the \( \varepsilon \)-Bias Detection problem of Corollary 12, with \( \varepsilon := \frac{1}{2\sqrt{N}} \). This \( C' \) will have depth \( d' = d + 2 \) and size \( S' = O(NS) \). By Corollary 12, this will imply that \( C' \) itself must have had size

\[
S = \exp \left( \Omega \left( \frac{1}{\varepsilon^{1/(d+2)}} \right) \right) = \exp \left( \Omega \left( \frac{1}{\varepsilon^{1/(2d+8)}} \right) \right).
\]

Let \( M = N^2n \), and let \( R = r_1 \ldots r_M \in \{0,1\}^M \) be a string of bits where each \( r_j \) is 1 with independent probability \( p \). We want to decide whether \( p = 1/2 \) or \( p = 1/2 + \varepsilon \)—that is, whether \( R \) was drawn from \( U[0] \) or \( U[\varepsilon] \). We can do this as follows. First, choose strings \( s_1, \ldots, s_n \in \{0,1\}^n \), bits \( b_1, \ldots, b_n \in \{0,1\} \), and an integer \( k \in [n] \) uniformly at random. Next, define Boolean functions \( f_1, \ldots, f_n : \{0,1\}^{n} \rightarrow \{-1,1\} \) using the first \( Nn \) bits of \( R \), like so:

\[
f_i(x) := (-1)^{r_i x + s_i x + b_i}.
\]

Finally, feed \( \langle f_1, \ldots, f_n \rangle \) as input to \( C \), and consider \( z_k \), the \( k^{th} \) output of \( C \) (discarding the other \( n - 1 \) outputs). We are interested in \( \Pr[z_k = s_k] \), where the probability is over \( R, s_1, \ldots, s_n, b_1, \ldots, b_n, \) and \( k \).

If \( p = 1/2 \), notice that \( f_1, \ldots, f_n \) are independent and uniformly random regardless of \( s_1, \ldots, s_n \). So \( C \) gets no information about \( s_k \), and \( \Pr[z_k = s_k] = 1/N \).

On the other hand, if \( p = 1/2 + \varepsilon \), then each \( f_i \) is drawn independently from the distribution \( D[\varepsilon] \) studied in Lemma 13. So by the lemma, for every \( i \in [n] \), if \( \beta \) then

\[
\Pr_{f_i}[z_i = s_i] \geq \frac{e^\beta + e^{-\beta}}{2\sqrt{eN}}.
\]

So assuming \( C \) succeeds (that is, \( \langle f_1, \ldots, f_n \rangle \in G \)), we have

\[
\Pr_{f_1, \ldots, f_n, k}[z_k = s_k] \geq \frac{1}{4} \left( \frac{e^2 + e^{-2}}{2\sqrt{eN}} \right) + \left( \frac{3}{4} - \frac{1}{4} \right) \left( \frac{e^1 + e^{-1}}{2\sqrt{eN}} \right) \geq \frac{1.038}{N}.
\]

So for a random \( \langle f_1, \ldots, f_n \rangle \) drawn according to \( D^n \),

\[
\Pr_{f_1, \ldots, f_n, k}[z_k = s_k] \geq 0.98 \left( \frac{1.038}{N} \right) \geq \frac{1.017}{N}.
\]

Notice that this is bounded above \( 1/N \) by a multiplicative constant.

Now we repeat the above experiment \( 1/N \) times. That is, for all \( j := 1 \) to \( N \), we generate Boolean functions \( f_{j1}, \ldots, f_{jn} : \{0,1\}^{n} \rightarrow \{-1,1\} \) by the same probabilistic procedure as before, but each time using a new \( Nn \)-bit substring of \( R_j \) of \( R \), as well as new \( s, b, \) and \( k \) values (denoted
for each positive integer $n$. So in forming $C$. We then apply $C$ to each $n$-tuple $(f_{j1}, \ldots, f_{jn})$. Let $z_j$ be the $k_j^{th}$ string that $C$ outputs when run on $(f_{j1}, \ldots, f_{jn})$. Then by the above, for each $j \in [N]$ we have

$$p = \frac{1}{2} \implies \Pr [z_j = s_{jk_j}] = \frac{1}{N},$$

$$p = \frac{1}{2} + \varepsilon \implies \Pr [z_j = s_{jk_j}] \geq \frac{1.017}{N}.$$}

Furthermore, these probabilities are independent across the different $j$'s. So let $E$ be the event that there exists a $j \in [N]$ such that $z_j = s_{jk_j}$. Then if $p = 1/2$ we have

$$\Pr [E] = 1 - \left(1 - \frac{1}{N}\right)^N \approx 1 - \frac{1}{e} \leq 0.633,$$

while if $p = 1/2 + \varepsilon$ we have

$$\Pr [E] \geq 1 - \left(1 - \frac{1.017}{N}\right)^N \geq 0.638.$$

We can now create the circuit $C'$, which distinguishes $R \in \{0, 1\}^M$ drawn from $U[0]$ from $R$ drawn from $U[\varepsilon]$ with constant bias. For each $j \in [N]$, generate an $n$-tuple of Boolean functions $(f_{j1}, \ldots, f_{jn})$ from $R$ and apply $C$ to it; then check whether there exists a $j \in [N]$ such that $z_j = s_{jk_j}$. This checking step can be done by a depth-2 circuit of size $O(Nn)$. Therefore, $C'$ will have depth $d' = d + 2$ and size $s' = O(Ns)$. A technicality is that our choices of the $s_{ji}$'s, $b_{ji}$'s, and $k_j$'s were made randomly. But by Yao’s principle, there certainly exist $s_{ji}$'s, $b_{ji}$'s, and $k_j$'s such that

$$\Pr_{U[\varepsilon]} [C' (R)] - \Pr_{U[0]} [C' (R)] \geq 0.638 - 0.633 = 0.005.$$}

So in forming $C'$, we simply hardwire those choices. ■

Combining Theorem 15 with standard diagonalization tricks, we can now prove an oracle separation (in fact, a random oracle separation) between the complexity classes $FBQP$ and $FBPP^{PH}$.

**Theorem 16** $FBQP^A \not\subseteq FBPP^{PH^A}$ with probability 1 for a random oracle $A$.

**Proof.** We interpret $A$ as encoding $n$ random Boolean functions $f_{n1}, \ldots, f_{nn} : \{0, 1\}^n \to \{-1, 1\}$ for each positive integer $n$. Let $R$ be the relational problem where we are given $0^n$ as input, and succeed if and only if we output strings $z_1, \ldots, z_n \in \{0, 1\}^n$, at least $3/4$ of which satisfy $|\hat{f}_{ni} (z_i)| \geq 1$ and at least $1/4$ of which satisfy $|\hat{f}_{ni} (z_i)| \geq 2$. Then by Lemmas 7 and 8, there exists an $FBQP^A$ machine $M$ such that for all $n$,

$$\Pr [M (0^n) \text{ succeeds}] \geq 1 - \frac{1}{\exp (n)},$$

where the probability is over both $A$ and the quantum randomness. Hence $\Pr [M (0^n) \text{ succeeds}] \geq 1 - 1/\exp (n)$ on all but finitely many $n$, with probability 1 over $A$. Since we can simply hardwire the answers on the $n$’s for which $M$ fails, it follows that $R \in FBQP^A$ with probability 1 over $A$.

On the other hand, let $M$ be an $FBPP^{PH^A}$ machine. Then by the standard conversion between $PH$ and $AC^0$, for every $n$ there exists a probabilistic $AC^0$ circuit $C_{M,n}$, of size $2^{\text{poly}(n)} = 2^{\text{polylog}(N)}$,
that takes \( A \) as input and simulates \( M(0^n) \). By Yao’s principle, we can assume without loss of generality that \( C_{M,n} \) is deterministic, since the oracle \( A \) is already random. Then by Theorem 15,

\[
\Pr_A [C_{M,n} \text{ succeeds}] < 0.99
\]

for all sufficiently large \( n \). By the independence of the \( f_{ni} \)'s, this is true even if we condition on \( C_{M,1}, \ldots, C_{M,n-1} \) succeeding. So as in the standard random oracle argument of Bennett and Gill [12], for every fixed \( M \) we have

\[
\Pr_A [C_{M,1}, C_{M,2}, C_{M,3}, \ldots \text{ all succeed}] = 0.
\]

So by the union bound,

\[
\Pr_A [\exists M : C_{M,1}, C_{M,2}, C_{M,3}, \ldots \text{ all succeed}] = 0
\]

as well. It follows that \( \text{FBQP}^A \not\subseteq \text{FBPP}^{PH^A} \) with probability 1 over \( A \).

If we “scale down by an exponential,” then we can eliminate the need for the oracle \( A \), and get a relation problem that is solvable in quantum logarithmic time but not in \( \text{AC}^0 \).

**Theorem 17** There exists a relation problem in \( \text{BQLOGTIME} \setminus \text{AC}^0 \).

**Proof.** In our relation problem \( R \), the input (of size \( M = 2^n n \)) will encode the truth tables of \( n \) Boolean functions, \( f_1, \ldots, f_n : \{0,1\}^n \to \{-1,1\} \), which are promised to be “good” as defined in Section 2.1. The task is to solve \( \text{Promise Fourier Fishing} \) on \( \langle f_1, \ldots, f_n \rangle \).

By Lemma 7, there exists a quantum algorithm that runs in \( O(n) = O(\log M) \) time, making random accesses to the truth tables of \( f_1, \ldots, f_n \), that solves \( R \) with probability \( 1 - 1/\exp(n) = 1 - 1/M^{\Omega(1)} \).

On the other hand, suppose \( R \) is in \( \text{AC}^0 \). Then we get a nonuniform circuit family \( \{C_n\}_n \), of depth \( O(1) \) and size \( \text{poly}(M) = 2^{O(n)} \), that solves \( \text{Fourier Fishing} \) on all tuples \( \langle f_1, \ldots, f_n \rangle \) that are good. Recall that by Lemma 8, a \( 1 - 1/\exp(n) \) fraction of \( \langle f_1, \ldots, f_n \rangle \)'s are good. Therefore \( \{C_n\}_n \) actually solves \( \text{Fourier Fishing} \) with probability \( 1 - 1/\exp(n) \) on \( \langle f_1, \ldots, f_n \rangle \) chosen uniformly at random. But this contradicts Theorem 15.

Hence \( R \in \text{FBQLOGTIME} \setminus \text{AC}^0 \) (where \( \text{FBQLOGTIME} \) and \( \text{AC}^0 \) are the relation versions of \( \text{BQLOGTIME} \) and \( \text{AC}^0 \) respectively).

## 5 The Classical Complexity of Fourier Checking

Section 4 settled the relativized \( \text{BQP} \) versus \( \text{PH} \) question, if we are willing to talk about relation problems. Ultimately, though, we also care about decision problems. So in this section we consider the \( \text{Fourier Checking} \) problem, of deciding whether two Boolean functions \( f, g \) are independent or forrelated. In Section 3.2, we saw that \( \text{Fourier Checking} \) has quantum query complexity \( O(1) \). What is its classical query complexity?\(^{12}\)

It is not hard to give a classical algorithm that solves \( \text{Fourier Checking} \) using \( O\left(\sqrt{N}\right) = O\left(2^{n/2}\right) \) queries. The algorithm is as follows: for some \( K = \Theta\left(\sqrt{N}\right) \), first choose sets \( X = \)

\(^{12}\)So long as we consider the distributional version of \( \text{Fourier Checking} \), the deterministic and randomized query complexities are the same (by Yao’s principle).
\{x_1, \ldots, x_K\} and \(Y = \{y_1, \ldots, y_K\}\) of \(n\)-bit strings uniformly at random. Then query \(f(x_i)\) and \(g(y_i)\) for all \(i \in [K]\). Finally, compute
\[
Z := \sum_{i,j=1}^{K} f(x_i) (-1)^{x_i \cdot y_j} g(y_j),
\]
accept if \(|Z|\) is greater than some cutoff \(cK\), and reject otherwise. For suitable \(K\) and \(c\), one can show that this algorithm accepts a forrelated \((f, g)\) pair with probability at least \(2/3\), and accepts a random \((f, g)\) pair with probability at most \(1/3\). We omit the details of the analysis, as they are tedious and not needed elsewhere in the paper.

In the next section, we will show that \textsc{Fourier Checking} has a property called \textit{almost \(k\)-wise independence}, which immediately implies a lower bound of \(\Omega \left( \sqrt[4]{N} \right) = \Omega \left( 2^{n/4} \right)\) on its randomized query complexity\(^{13}\) (as well as exponential lower bounds on its \textsc{MA}, \textsc{BPP}_{\text{path}}, and \textsc{SZK} query complexities). Indeed, we conjecture that almost \(k\)-wise independence is enough to imply that \textsc{Fourier Checking} is not in \textsc{PH}. We discuss the status of that conjecture in Section 6.

### 5.1 Almost \(k\)-Wise Independence

Let \(X = x_1 \ldots x_N \in \{0, 1\}^N\) be a string. Then a \textit{literal} is an expression of the form \(x_i\) or \(1 - x_i\), and a \textit{k-term} is a product of \(k\) literals (each involving a different \(x_i\)), which is \(1\) if the literals all take on prescribed values and \(0\) otherwise. We can give an analogous definition for strings over the alphabet \((-1, 1)\); we simply let the literals have the form \(\frac{1 \pm x_i}{2}\).

Let \(U\) be the uniform distribution over \(N\)-bit strings. The following definition will play a major role in this work.

\textbf{Definition 18} A distribution \(D\) over \(N\)-bit strings is \(\varepsilon\)-almost \(k\)-wise independent if for every \(k\)-term \(C\),
\[
1 - \varepsilon \leq \frac{\Pr_{X \sim D}[C(X)]}{\Pr_{X \sim U}[C(X)]} \leq 1 + \varepsilon.
\]
(Note that \(\Pr_{X \sim U}[C(X)]\) is just \(2^{-k}\).)

In other words, there should be no assignment to any \(k\) input bits, such that conditioning on that assignment gives us much information about whether \(X\) was drawn from \(D\) or \(U\).

Now let \(F\) be the forrelated distribution over pairs of Boolean functions \(f, g : \{0, 1\}^n \to \{-1, 1\}\), where we first choose a vector \(v = (v_x)_{x \in \{0, 1\}^n} \in \mathbb{R}^N\) of independent \(\mathcal{N}(0, 1)\) Gaussians, then set \(f(x) := \text{sgn}(v_x)\) and \(g(x) := \text{sgn}(\tilde{v}_x)\) for all \(x \in \{0, 1\}^n\).

\textbf{Theorem 19} \(F\) is \(O \left( \frac{k^2}{\sqrt{N}} \right)\)-almost \(k\)-wise independent for all \(k \leq \sqrt[4]{N}\).

\textbf{Proof.} As a first step, we will prove an analogous statement for the real-valued functions \(F(x) := v_x\) and \(G(y) := \tilde{v}_y\); then we will generalize to the discrete versions \(f(x)\) and \(g(y)\). Let \(U'\) be the probability measure over \((F, G)\) that corresponds to case (i) of \textsc{Fourier Checking}; that is, we

\(^{13}\)With some more work, we believe that the randomized query complexity lower bound can be improved to \(\Omega \left( \sqrt[4]{N} \right)\), though we have not pursued that here.
choose each $F(x)$ and $G(x)$ value independently from the Gaussian measure $\mathcal{N}(0,1)$. Let $\mathcal{F}'$ be the probability measure over $(F,G)$ that corresponds to case (ii) of Fourier Checking: that is, we choose each $F(x)$ independently from $\mathcal{N}(0,1)$, and then set $G(y) := \hat{F}(y)$ where

$$\hat{F}(y) = \frac{1}{\sqrt{N}} \sum_{x \in \{0,1\}^n} (-1)^x y F(x)$$

is the Fourier transform of $F$. Observe that since the Fourier transform is unitary, $G$ has the same marginal distribution as $F$ under $\mathcal{F}'$: namely, a product of independent $\mathcal{N}(0,1)$ Gaussians.

Fix inputs $x_1, \ldots, x_K \in \{0,1\}^n$ of $F$ and $y_1, \ldots, y_L \in \{0,1\}^n$ of $G$, for some $K, L \leq N^{1/4}$. Then given constants $a_1, \ldots, a_K, b_1, \ldots, b_L \in \mathbb{R}$, let $S$ be the set of all $(F,G)$ that satisfy the $K+L$ equations

$$F(x_i) = a_i \text{ for } 1 \leq i \leq K,$$

$$G(y_j) = b_j \text{ for } 1 \leq j \leq L.$$  \hfill (1)

We can think of $S$ as a $(2N-K-L)$-dimensional affine subspace in $\mathbb{R}^{2N}$. The measure of $S$ under some probability measure $\mu$ on $\mathbb{R}^{2N}$, is simply

$$\mu(S) := \int_{(F,G) \in S} \mu(F,G) d(F,G).$$

Now let

$$\Delta_S := a_1^2 + \cdots + a_K^2 + b_1^2 + \cdots + b_L^2$$

be the squared distance between $S$ and the origin (that is, the minimum squared 2-norm of any point in $S$). Then by the spherical symmetry of the Gaussian measure, it is not hard to see that $S$ has measure

$$\mathcal{U}'(S) = \frac{e^{-\Delta_S/2}}{\sqrt{2\pi}^{K+L}}$$

under $\mathcal{U}'$. Our key claim is that

$$1 - O \left( \frac{(K+L)\Delta_S}{\sqrt{N}} \right) \leq \frac{\mathcal{F}'(S)}{\mathcal{U}'(S)} \leq 1 + O \left( \frac{(K+L)\Delta_S}{\sqrt{N}} \right).$$

To prove this claim: recall that the probability measure over $F$ induced by $\mathcal{F}'$ is just a spherical Gaussian $\mathcal{G}$ on $\mathbb{R}^N$, and that $G = \hat{F}$ uniquely determines $F$ and vice versa. So consider the $(N-K-L)$-dimensional affine subspace $T$ of $\mathbb{R}^N$ defined by the $K+L$ equations

$$F(x_i) = a_i \text{ for } 1 \leq i \leq K,$$

$$\hat{F}(y_j) = b_j \text{ for } 1 \leq j \leq L.$$  \hfill (2)

Then $\mathcal{F}'(S) = \mathcal{G}(T)$: that is, to compute how much measure $\mathcal{F}'$ assigns to $S$, it suffices to compute how much measure $\mathcal{G}$ assigns to $T$. We have

$$\mathcal{G}(T) = \frac{e^{-\Delta_T/2}}{\sqrt{2\pi}^{K+L}}.$$
where $\Delta_T$ is the squared Euclidean distance between $T$ and the origin. Thus, our problem reduces to minimizing
\[
\Delta_F := \sum_{x \in \{0,1\}^n} F(x)^2
\]
over all $F \in T$. By a standard fact about quadratic optimization, the optimal $F \in T$ will have the form
\[
F(x) = \alpha_1 E_1(x) + \cdots + \alpha_K E_K(x) + \beta_1 \chi_1(x) + \cdots + \beta_L \chi_L(x)
\]
where
\[
E_i(x) := \begin{cases} 1 & \text{if } x = x_i \\ 0 & \text{otherwise} \end{cases}
\]
is an indicator function, and
\[
\chi_j(x) := \frac{(-1)^x y_j}{\sqrt{N}}
\]
is the $y^j$th Fourier character evaluated at $x$. Furthermore, the coefficients $\{\alpha_i\}_{i \in [K]}, \{\beta_j\}_{j \in [L]}$ can be obtained by solving the linear system
\[
\begin{pmatrix}
1 & 0 & 0 & \pm 1/\sqrt{N} & \cdots & \pm 1/\sqrt{N} \\
0 & \ddots & 0 & \vdots & \ddots & \vdots \\
\pm 1/\sqrt{N} & \cdots & \pm 1/\sqrt{N} & 1 & 0 & 0 \\
\vdots & \ddots & \vdots & 0 & \ddots & \vdots \\
\pm 1/\sqrt{N} & \cdots & \pm 1/\sqrt{N} & 0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
\alpha_1 \\
\vdots \\
\alpha_K \\
\beta_1 \\
\vdots \\
\beta_L
\end{pmatrix}
= \begin{pmatrix}
a_1 \\
\vdots \\
a_K \\
b_1 \\
\vdots \\
b_L
\end{pmatrix}
\]
\[
A
\]
Here $A$ is simply a matrix of covariances: the top left block records the inner product between each $E_i$ and $E_j$ (and hence is a $K \times K$ identity matrix), the bottom right block records the inner product between each $\chi_i$ and $\chi_j$ (and hence is an $L \times L$ identity matrix), and the remaining two blocks of size $K \times L$ record the inner product between each $E_i$ and $\chi_j$.

Thus, to get the vector of coefficients $u \in \mathbb{R}^{K+L}$, we simply need to calculate $A^{-1}w$. Let $B = I - A$. Then by Taylor series expansion,
\[
A^{-1} = (I - B)^{-1} = I + B + B^2 + B^3 + \cdots
\]
Notice that every entry of $B$ is at most $1/\sqrt{N}$ in absolute value. This means that, for all positive integers $t$, every entry of $B^t$ is at most
\[
\frac{(K + L)^{t-1}}{N^{t/2}}
\]
in absolute value. Since $K + L \ll \sqrt{N}$, this in turn means that every entry of $I - A^{-1}$ has absolute value $O\left(1/\sqrt{N}\right)$. So $A^{-1}$ is exponentially close to the identity matrix. Hence, when we compute the vector $u = A^{-1}w$, we find that
\[
\alpha_i = a_i + \varepsilon_i \text{ for all } 1 \leq i \leq K,
\]
\[
\beta_j = b_j + \delta_j \text{ for all } 1 \leq j \leq L,
\]
for some small error terms $\varepsilon_i$ and $\delta_j$. Specifically, each $\varepsilon_i$ and $\delta_j$ is the inner product of $w$, a $(K + L)$-dimensional vector of length $\sqrt{\Delta S}$, with a vector every entry of which has absolute value $O\left(1/\sqrt{N}\right)$. By Cauchy-Schwarz, this implies that

$$|\varepsilon_i|, |\delta_j| = O\left(\sqrt{(K + L) \Delta S / N}\right)$$

for all $i, j$. So

$$\Delta_T = \min_{F \in T} \sum_{x \in \{0, 1\}^n} F(x)^2$$

$$= \sum_{i=1}^K \alpha_i^2 + \sum_{j=1}^L \beta_j^2 + 2 \sum_{i=1}^K \sum_{j=1}^L \frac{\alpha_i \beta_j}{\sqrt{N}}$$

$$= \sum_{i=1}^K (a_i + \varepsilon_i)^2 + \sum_{j=1}^L (b_j + \delta_j)^2 + 2 \sum_{i=1}^K \sum_{j=1}^L \frac{(a_i + \varepsilon_i)(b_j + \delta_j)}{\sqrt{N}}$$

$$= \Delta_S \pm O\left(\frac{(K + L) \Delta S}{\sqrt{N}} + \frac{(K + L)^2 \Delta S}{N} + \frac{(K + L)^3 \Delta S}{N^{3/2}}\right)$$

$$= \Delta_S \left(1 \pm O\left(\frac{K + L}{\sqrt{N}}\right)\right),$$

where the fourth line made repeated use of Cauchy-Schwarz, and the fifth line used the fact that $K + L \ll \sqrt{N}$. Hence

$$\frac{\mathcal{F}'(S)}{\mathcal{U}(S)} = e^{-\Delta_T/2} / \sqrt{2\pi}^{K+L}$$

$$= e^{-\Delta_S^2/2} / \sqrt{2\pi}^{K+L}$$

$$= \exp\left(\frac{\Delta_S - \Delta_T}{2}\right)$$

$$= \exp\left(\pm O\left(\frac{(K + L) \Delta S}{\sqrt{N}}\right)\right)$$

$$= 1 \pm O\left(\frac{(K + L) \Delta S}{\sqrt{N}}\right)$$

which proves the claim.

To prove the theorem, we now need to generalize to the discrete functions $f$ and $g$. Here we are given a term $C$ that is a conjunction of $K + L$ inequalities: $K$ of the form $F(x_i) \leq 0$ or $F(x_i) \geq 0$, and $L$ of the form $G(y_j) \leq 0$ or $G(y_j) \geq 0$. If we fix $x_1, \ldots, x_K$ and $y_1, \ldots, y_L$, we can think of $C$ as just a convex region of $\mathbb{R}^{K+L}$. Then given an affine subspace $S$ as defined by equation (1), we will (abusing notation) write $S \in C$ if the vector $(\alpha_1, \ldots, \alpha_K, \beta_1, \ldots, \beta_L)$ is in $C$: that is, if $S$ is compatible with the $K + L$ inequalities that define $C$. We need to show that the ratio
\[
\frac{\text{Pr}_F[C]}{\text{Pr}_U[C]} \approx 1. \quad \text{We can do so using the previous result, as follows:}
\]

\[
\frac{\text{Pr}_F[C]}{\text{Pr}_U[C]} = \int_{S \in C} \mathcal{F}'(S) \frac{dS}{\int_{S \in C} \mathcal{U}'(S) \, dS}
\]

\[
= \int_{S \in C} \mathcal{U}'(S) \left[ 1 \pm O \left( \frac{(K+L)\Delta S}{\sqrt{N}} \right) \right] \, dS
\]

\[
= \frac{\int_{S \in C} \mathcal{U}'(S) \left[ e^{-\Delta S/2 \sqrt{2\pi K+L}} \right] \left[ 1 \pm O \left( \frac{(K+L)\Delta S}{\sqrt{N}} \right) \right] \, dS}{\int_{S \in C} \mathcal{U}'(S) \, dS}
\]

\[
= \frac{(1/2)^{K+L} \pm O \left( \int_{S \in C} \left[ e^{-\Delta S/2 \sqrt{2\pi K+L}} \right] \frac{(K+L)\Delta S}{\sqrt{N}} \, dS \right)}{(1/2)^{K+L}}
\]

\[
= 1 \pm \frac{2^{K+L} (K+L)}{\sqrt{N}} O \left( \int_{S \in C} \frac{e^{-\Delta S/2 \sqrt{2\pi K+L} \Delta S}}{\sqrt{N}} \, dS \right)
\]

\[
= 1 \pm \frac{K+L}{\sqrt{N}} O \left( \int_{S \sqrt{2\pi K+L} \Delta S} \frac{e^{-\Delta S/2 \sqrt{2\pi K+L} \Delta S}}{\sqrt{N}} \, dS \right)
\]

\[
= 1 \pm O \left( \frac{(K+L)^2}{\sqrt{N}} \right).
\]

Setting \( k := K + L \), this completes the proof. \( \blacksquare \)

### 5.2 Oracle Separation Results

The following lemma shows that any almost \( k \)-wise independent distribution is indistinguishable from the uniform distribution by \( \text{BPP}_{\text{path}} \) or \( \text{SZK} \) machines.

**Lemma 20** Suppose a probability distribution \( D \) over oracle strings is \( 1/t(n) \)-almost \( \text{poly} (n) \)-wise independent, for some superpolynomial function \( t \). Then no \( \text{BPP}_{\text{path}} \) machine or \( \text{SZK} \) protocol can distinguish \( D \) from the uniform distribution \( U \) with non-negligible bias.

**Proof.** Let \( M \) be a \( \text{BPP}_{\text{path}} \) machine, and let \( p_D \) be the probability that \( M \) accepts an oracle string drawn from distribution \( D \). Then \( p_D \) can be written as \( a_D/s_D \), where \( s_D \) is the fraction of \( M \)'s computation paths that are postselected, and \( a_D \) is the fraction of \( M \)'s paths that are both postselected and accepting. Since each computation path can examine at most \( \text{poly} (n) \) bits and \( D \) is \( 1/t(n) \)-almost \( \text{poly} (n) \)-wise independent, we have

\[
1 - \frac{1}{t(n)} \leq \frac{a_D}{s_D} \leq 1 + \frac{1}{t(n)} \quad \text{and} \quad 1 - \frac{1}{t(n)} \leq \frac{s_D}{s_U} \leq 1 + \frac{1}{t(n)}.
\]

Hence

\[
\left( 1 - \frac{1}{t(n)} \right)^2 \leq \frac{a_D/s_D}{a_U/s_U} \leq \left( 1 + \frac{1}{t(n)} \right)^2.
\]

Now let \( P \) be an \( \text{SZK} \) protocol. Then by a result of Sahai and Vadhan [34], there exist polynomial-time samplable distributions \( A \) and \( A' \) such that if \( P \) accepts, then \( \| A - A' \| \leq 1/3 \),
while if $P$ rejects, then $\|A - A'\| \geq 2/3$. But since each computation path can examine at most poly($n$) oracle bits and $D$ is $1/t(n)$-almost poly($n$)-wise independent, we have $\|A_D - A'_D\| \leq 1/t(n)$ and $\|A'_D - A'_U\| \leq 1/t(n)$, where the subscript denotes the distribution from which the oracle string was drawn. Hence
\[
\|A_D - A'_D\| - \|A_U - A'_U\| \leq \|A_D - A_U\| + \|A'_D - A'_U\| \leq \frac{2}{t(n)}
\]
and no $\text{SZK}$ protocol exists. \hfill $\blacksquare$

We now combine Theorem 19 and Lemma 20 with standard diagonalization tricks, to obtain an oracle relative to which $\text{BQP} \not\subseteq \text{BPP}_{\text{path}}$ and $\text{BQP} \not\subseteq \text{SZK}$.

**Theorem 21.** There exists an oracle $A$ relative to which $\text{BQP}^A \not\subseteq \text{BPP}^A_{\text{path}}$ and $\text{BQP}^A \not\subseteq \text{SZK}^A$.

**Proof.** The oracle $A$ will encode the truth tables of Boolean functions $f_1, f_2, \ldots$ and $g_1, g_2, \ldots$, where $f_n, g_n : \{0, 1\}^n \rightarrow \{-1, 1\}$ are on $n$ variables each. For each $n$, with probability $1/2$ we draw $\langle f_n, g_n \rangle$ from the uniform distribution $\mathcal{U}$, and with probability $1/2$ we draw $\langle f_n, g_n \rangle$ from the forrelated distribution $\mathcal{F}$. Let $L$ be the unary language consisting of all $0^n$ for which $\langle f_n, g_n \rangle$ was drawn from $\mathcal{F}$.

By Theorem 9, there exists a $\text{BQP}^A$ machine $M$ that decides $L$ on all but finitely many values of $n$, with probability 1 over $A$. Since we can simply hardwire the values of $n$ on which $M$ fails, it follows that $L \in \text{BQP}^A$ with probability 1 over $A$.

On the other hand, we showed in Theorem 19 that $\mathcal{F}$ is $O\left(\frac{p(n)^2}{2^{n/2}}\right)$-almost $p(n)$-wise independent for all polynomials $p$. Hence, by Lemma 20, no $\text{BPP}_{\text{path}}$ machine can distinguish $\mathcal{F}$ from $\mathcal{U}$ with non-negligible bias. Let $E_n(M)$ be the event that the $\text{BPP}^A_{\text{path}}$ machine $M$ correctly decides whether $0^n \in L$. Then
\[
\Pr_M[ E_n(M) ] \leq \frac{1}{2} + o(1),
\]
and moreover this is true even conditioning on $E_1(M), \ldots, E_{n-1}(M)$. So as in a standard random oracle argument [12], for every fixed $M$ we have
\[
\Pr_M[ E_1(M) \land E_2(M) \land \cdots ] = 0.
\]
So by the union bound,
\[
\Pr_A[ \exists M : E_1(M) \land E_2(M) \land \cdots ] = 0
\]
as well. It follows that $\text{BQP}^A \not\subseteq \text{BPP}^A_{\text{path}}$ with probability 1 over $A$. By exactly the same argument, we also get $\text{BQP}^A \not\subseteq \text{SZK}^A$ with probability 1 over $A$. \hfill $\blacksquare$

Since $\text{BPP} \subseteq \text{MA} \subseteq \text{BPP}_{\text{path}}$, Theorem 21 supersedes the previous results that there exist oracles $A$ relative to which $\text{BPP}^A \neq \text{BQP}^A$ [13] and $\text{BQP}^A \not\subseteq \text{MA}^A$ [42].

### 6 The Generalized Linial-Nisan Conjecture

In 1990, Linial and Nisan [31] conjectured that “polylogarithmic independence fools AC$^0$”—or loosely speaking, that every probability distribution $\mathcal{D}$ over $n$-bit strings that is uniform on all small subsets of bits, is indistinguishable from the uniform distribution by polynomial-size, constant-depth circuits. We now state a variant of the Linial-Nisan Conjecture, not with the best parameters but with weaker, easier-to-understand parameters that suffice for our application.
Conjecture 22 (Linial-Nisan or LN Conjecture) Let $\mathcal{D}$ be an $n^{\Omega(1)}$-wise independent distribution over $\{0,1\}^n$, and let $f : \{0,1\}^n \rightarrow \{0,1\}$ be computed by an $\text{AC}^0$ circuit of size $2^{n^{o(1)}}$ and depth $O(1)$. Then
\[
\left| \Pr_{x \sim \mathcal{D}}[f(x)] - \Pr_{x \sim \mathcal{U}}[f(x)] \right| = o(1).
\]

After seventeen years of almost no progress, in 2007 Bazzi [8] finally proved the LN Conjecture for the special case of depth-2 circuits (also called DNF formulas). Bazzi’s proof was about 50 pages, but it was dramatically simplified a year later, when Razborov [33] discovered a 3-page proof. Then in 2009, Braverman [15] gave a breakthrough proof of the full LN Conjecture.

Theorem 23 (Braverman’s Theorem [15]) Let $f : \{0,1\}^n \rightarrow \{0,1\}$ be computed by an $\text{AC}^0$ circuit of size $S$ and depth $d$, and let $\mathcal{D}$ be a $(\log \frac{S}{\varepsilon})^{7d^2}$-wise independent distribution over $\{0,1\}^n$. Then for all sufficiently large $S$,
\[
\left| \Pr_{x \sim \mathcal{D}}[f(x)] - \Pr_{x \sim \mathcal{U}}[f(x)] \right| \leq \varepsilon.
\]

We conjecture a modest-seeming extension of Braverman’s Theorem, which says (informally) that almost $k$-wise independent distributions fool $\text{AC}^0$ as well.

Conjecture 24 (Generalized Linial-Nisan or GLN Conjecture) Let $\mathcal{D}$ be a $1/n^{\Omega(1)}$-almost $n^{\Omega(1)}$-wise independent distribution over $\{0,1\}^n$, and let $f : \{0,1\}^n \rightarrow \{0,1\}$ be computed by an $\text{AC}^0$ circuit of size $2^{n^{o(1)}}$ and depth $O(1)$. Then
\[
\left| \Pr_{x \sim \mathcal{D}}[f(x)] - \Pr_{x \sim \mathcal{U}}[f(x)] \right| = o(1).
\]

By the usual correspondence between $\text{AC}^0$ and $\text{PH}$, the GLN Conjecture immediately implies the following counterpart of Lemma 20:

Suppose a probability distribution $\mathcal{D}$ over oracle strings is $1/n^{\Omega(1)}$-almost $\text{poly}(n)$-wise independent, for some superpolynomial function $t$. Then no $\text{PH}$ machine can distinguish $\mathcal{D}$ from the uniform distribution $\mathcal{U}$ with non-negligible bias.

And thus we get the following implication:

Theorem 25 Assuming the GLN Conjecture, there exists an oracle $A$ relative to which $\text{BQP}^A \not\subset \text{PH}^A$.

Proof. The proof is the same as that of Theorem 21; the only difference is that the GLN Conjecture now plays the role of Lemma 20. ■

Likewise:

Theorem 26 Assuming the GLN Conjecture for the special case of depth-2 circuits (i.e., DNF formulas), there exists an oracle $A$ relative to which $\text{BQP}^A \not\subset \text{AM}^A$. 

**Proof.** Just like in Theorem 21, define an oracle $A$ and an associated language $L$ using the **Fourier Checking** problem. Then $L \in \text{BQP}^A$ with probability 1 over the choices made in constructing $A$. On the other hand, suppose $L \in \text{AM}^A$ with probability 1 over $A$. Then we claim that **Fourier Checking** can also be solved by a family of DNF formulas $\{\varphi_n\}_{n \geq 1}$ of size $2^{\text{poly}(n)}$:

$$\left| \Pr_{(f, g) \sim \mathcal{F}} [\varphi_n (f, g)] - \Pr_{(f, g) \sim U} [\varphi_n (f, g)] \right| = \Omega (1).$$

But since $\mathcal{F}$ is $O (k^2/2^n)$-almost $k$-wise independent (by Theorem 19), such a family $\varphi_n$ would violate the depth-2 case of the GLN Conjecture.

We now prove the claim. For simplicity, fix an input length $n$, and let $A$ refer to a single instance $⟨f, g⟩$ of **Fourier Checking**.

14 Let $P$ be an AM protocol that successfully distinguishes the forrelated distribution $\mathcal{F}$ over $A$ from the uniform distribution $U$. We can assume without loss of generality that $P$ is public-coin [26]. In other words, Arthur first sends a random challenge $r \in \{0, 1\}^{\text{poly}(n)}$ to Merlin, then Merlin responds with a witness $w \in \{0, 1\}^{\text{poly}(n)}$, then Arthur runs a deterministic polynomial-time verification procedure $V^A (r, w)$ to decide whether to accept. By the assumption that $P$ succeeds,

$$\left| \Pr_{A \sim D, r} [\exists w : V^A (r, w)] - \Pr_{A \sim D, r} [\exists w : V^A (r, w)] \right| = \Omega (1).$$

So by Yao’s principle, there exists a **fixed** challenge $r^*$ such that

$$\left| \Pr_{A \sim D} [\exists w : V^A (r^*, w)] - \Pr_{A \sim D} [\exists w : V^A (r^*, w)] \right| = \Omega (1).$$

Now let $Q_{A, w}$ be the set of all queries that $V^A (r^*, w)$ makes to $A$, and let $C_{A, w} (A')$ be a term (i.e., a conjunction of 1’s and 0’s) that returns TRUE if and only if $A'$ agrees with $A$ on all queries in $Q_{A, w}$. Then we can assume without loss of generality that $C_w := C_{A, w}$ depends only on $w$, not on $A$—since Merlin can simply tell Arthur which queries $V$ is going to make and what their outcomes will be, and Arthur can reject if Merlin is lying. Let $W$ be the set of all witnesses $w$ such that Arthur accepts if $C_w (A)$ returns TRUE. Consider the DNF formula

$$\varphi (A) := \bigvee_{w \in W} C_w (A),$$

which says that there exists a $w$ causing $V^A (r^*, w)$ to accept. Then $\varphi$ contains $2^{\text{poly}(n)}$ terms with $\text{poly}(n)$ literals each, and

$$\left| \Pr_{A \sim D} [\varphi (A)] - \Pr_{A \sim D} [\varphi (A)] \right| = \Omega (1).$$

As a side note, it is conceivable that one could prove

$$\Pr_{x \sim D} [\varphi (x)] - \Pr_{x \sim U} [\varphi (x)] = o (1).$$

14 It is straightforward to generalize to the case where Arthur can query other instances, besides the one he is trying to solve.
for every almost $k$-wise independent distribution $D$ and small CNF formula $\varphi$, without getting the same result for DNF formulas (or vice versa). However, since BQP is closed under complement, even such an asymmetric result would imply an oracle $A$ relative to which $\text{BQP}^A \not\subseteq \text{AM}^A$.

If the GLN Conjecture holds, then we can also “scale down by an exponential,” to obtain an unrelativized decision problem that is solvable in quantum logarithmic time but not in $\text{AC}^0$.

**Theorem 27** Assuming the GLN Conjecture, there exists a promise problem in $\text{BQLOGTIME}\setminus \text{AC}^0$.

**Proof.** In our promise problem $\Pi = (\Pi_{\text{YES}}, \Pi_{\text{NO}})$, the inputs (of size $M = 2^{n+1}$) will encode pairs of Boolean functions $f, g : \{0, 1\}^n \to \{-1, 1\}$, such that

$$p(f, g) := \frac{1}{N^3} \left( \sum_{x,y \in \{0,1\}^n} f(x)(-1)^{x \cdot y}g(y) \right)^2$$

is either at least 0.05 or at most 0.01. The problem is to accept in the former case and reject in the latter case.

Using the algorithm $\text{FC-ALG}$ from Section 3.2, it is immediate that $\Pi \in \text{BQLOGTIME}$. On the other hand, suppose $\Pi \in \text{AC}^0$. Then we get a nonuniform circuit family $\{C_n\}_n$ of depth $O(1)$ and size $\text{poly}(M) = 2^{O(n)}$, that solves $\text{FOURIER CHECKING}$ on all pairs $f, g$ such that (i) $p(f, g) \leq 0.01$ or (ii) $p(f, g) \geq 0.05$. By Corollary 10, the class (i) includes a $1 - 1/\exp(n)$ fraction of $(f, g)$’s drawn from the uniform distribution $U$, while the class (ii) includes a constant fraction of $(f, g)$’s drawn from the forrelated distribution $F$. Therefore, we actually obtain an $\text{AC}^0$ circuit family that distinguishes $U$ from $F$ with constant bias. But this contradicts Theorem 19 together with the GLN Conjecture. ■

### 6.1 Low-Fat Polynomials

Given that the GLN Conjecture would have such remarkable implications for quantum complexity theory, the question arises of how we can go about proving it. As we are indebted to Louay Bazzi for pointing out to us, the GLN Conjecture is equivalent to the following conjecture, about approximating $\text{AC}^0$ functions by low-degree polynomials.

**Conjecture 28** (Low-Fat Sandwich Conjecture) For every function $f : \{0, 1\}^n \to \{0, 1\}$ computable by an $\text{AC}^0$ circuit, there exist polynomials $p_\ell, p_u : \mathbb{R}^n \to \mathbb{R}$ of degree $k = n^{o(1)}$ that satisfy the following three conditions.

(i) **Sandwiching:** $p_\ell(x) \leq f(x) \leq p_u(x)$ for all $x \in \{0, 1\}^n$.

(ii) **$L_1$-Approximation:** $\mathbb{E}_{x \sim U}[p_u(x) - p_\ell(x)] = o(1)$.

(iii) **Low-Fat:** $p_\ell(x)$ and $p_u(x)$ can be written as linear combinations of terms, $p_\ell(x) = \sum_C \alpha_C C(x)$ and $p_u(x) = \sum_C \beta_C C(x)$ respectively, such that $\sum_C |\alpha_C|2^{-|C|} = n^{o(1)}$ and $\sum_C |\beta_C|2^{-|C|} = n^{o(1)}$. (Here a term $C$ is a product of literals of the form $x_i$ and $1-x_i$.)

If we take out condition (iii), then Conjecture 28 becomes equivalent to the original LN Conjecture (see Bazzi [8] for a proof). And indeed, all progress so far on “Linial-Nisan problems” has crucially relied on this connection with polynomials. Bazzi [8] and Razborov [33] proved the
depth-2 case of the LN Conjecture by constructing low-degree, approximating, sandwiching polynomials for every DNF, while Braverman [15] proved the full LN Conjecture by constructing such polynomials for every AC$^0$ circuit.\footnote{Strictly speaking, Braverman constructed approximating polynomials with slightly different (though still sufficient) properties. We know from Bazzi [8] that it must be possible to get sandwiching polynomials as well.} Given this history, proving Conjecture 28 would seem like the “obvious” approach to proving the GLN Conjecture.

Below we prove one direction of the equivalence: that to prove the GLN Conjecture, it suffices to construct low-fat sandwiching polynomials for every AC$^0$ circuit. The other direction—that the GLN Conjecture implies Conjecture 28, and hence, there is no loss of generality in working with polynomials instead of probability distributions—follows from a linear programming duality calculation that we omit.

**Theorem 29** The Low-Fat Sandwich Conjecture implies the GLN Conjecture.

**Proof.** Given an AC$^0$ function $f$, let $p_l, p_u$ be the low-fat sandwiching polynomials of degree $k$ that are guaranteed by Conjecture 28. Also, let $D$ be an $\varepsilon$-almost $k$-wise independent distribution over \{0, 1\}$^n$, for some $\varepsilon = 1/n^{\Omega(1)}$. Then

$$
\Pr_{x \sim D}[f(x)] - \Pr_{x \sim U}[f(x)] \leq \mathbb{E}_D[p_u] - \mathbb{E}_U[p_l]
\leq \sum_C \beta_C \mathbb{E}_D[C] - \mathbb{E}_U[p_l]
\leq \sum_C \frac{\beta_C + |\beta_C| \varepsilon}{2|C|} - \mathbb{E}_U[p_l]
= \mathbb{E}_U[p_u - p_l] + \varepsilon \sum_C |\beta_C| 2|C|
= \mathbb{E}_U[p_u - p_l] + \frac{n^{\Omega(1)}}{n^{\Omega(1)}}
= o(1).
$$

Likewise,

$$
\Pr_{x \sim U}[f(x)] - \Pr_{x \sim D}[f(x)] \leq \mathbb{E}_U[p_u] - \mathbb{E}_D[p_l]
\leq \mathbb{E}_U[p_u] - \sum_C \alpha_C \mathbb{E}_D[C]
\leq \mathbb{E}_U[p_u] - \sum_C \frac{\alpha_C - |\alpha_C| \varepsilon}{2|C|}
= \mathbb{E}_U[p_u - p_l] + \varepsilon \sum_C |\alpha_C| 2|C|
= o(1).
$$


7 Discussion

We now take a step back, and use our results to address some conceptual questions about the relativized BQP versus PH question, the GLN Conjecture, and what makes them so difficult.

The first question is an obvious one. Complexity theorists have known for decades how to prove constant-depth circuit lower bounds, and how to use those lower bounds to give oracles \( A \) relative to which (for example) \( \text{PP}^A \not\subset \text{PH}^A \) and \( \oplus^P \not\subset \text{PH}^A \). So why should it be so much harder to give an \( A \) relative to which \( \text{BQP}^A \not\subset \text{PH}^A \)? What makes this \( \text{AC}^0 \) lower bound different from all other \( \text{AC}^0 \) lower bounds?

The answer seems to be that, while we have powerful techniques for proving that a function \( f \) is not in \( \text{AC}^0 \), all of those techniques, in one way or another, involve arguing that \( f \) is not approximated by a low-degree polynomial. The Razborov-Smolensky technique \([32, 39]\) argues this explicitly, while even the random restriction technique \([19, 43, 40]\) argues it “implicitly,” as shown by Linial, Mansour, and Nisan \([30]\). And this is a problem, if \( f \) is also computed by an efficient quantum algorithm. For Beals et al. \([9]\) proved the following in 1998:

**Lemma 30** ([9]) Suppose a quantum algorithm \( Q \) makes \( T \) queries to a Boolean input \( X \in \{0,1\}^N \). Then \( Q \)'s acceptance probability is a real multilinear polynomial \( p(X) \), of degree at most \( 2T \).

In other words, if a function \( f \) is in \( \text{BQP} \), then for that very reason, \( f \) has a low-degree approximating polynomial! As an example, we already saw that the following polynomial \( p \), of degree 4, successfully distinguishes the forrelated distribution \( \mathcal{F} \) from the uniform distribution \( \mathcal{U} \):

\[
p(f,g) := \frac{1}{N^3} \left( \sum_{x,y \in \{0,1\}^n} f(x)(-1)^{x \cdot y} g(y) \right)^2 .
\]

Therefore, we cannot hope to prove a lower bound for Fourier Checking, by any argument that would also imply that such a \( p \) cannot exist.

This brings us to a second question. If

(i) every known technique for proving \( f \not\in \text{AC}^0 \) involves showing that \( f \) is not approximated by a low-degree polynomial, but

(ii) every function \( f \) with low quantum query complexity is approximated by a low-degree polynomial,

does that mean there is no hope of solving the relativized BQP versus PH problem using polynomial-based techniques?

We believe the answer is no. The essential point here is that an \( \text{AC}^0 \) function can be approximated by different kinds of low-degree polynomials. For example, Linial, Mansour, and Nisan \([30]\) showed that, if \( f : \{0,1\}^n \to \{0,1\} \) is in \( \text{AC}^0 \), then there exists a real polynomial \( p: \mathbb{R}^n \to \mathbb{R} \), of degree \( \text{polylog} \, n \), such that

\[
E_{x \in \{0,1\}^n} [(p(x) - f(x))^2] = o(1) .
\]
By comparison, Razborov [32] and Smolensky [39] showed that if \( f \in \text{AC}^0 \), then there exists a polynomial \( p : \mathbb{F}^n \to \mathbb{F} \) over any field \( \mathbb{F} \) (finite or infinite), of degree \( \text{polylog} N \), such that
\[
\Pr_{x \in \{0,1\}^n} [p(x) \neq f(x)] = o(1).
\]
Furthermore, to show that \( f \notin \text{AC}^0 \), it suffices to show that \( f \) is not approximated by a low-degree polynomial in any one of these senses. For example, even though the Parity function has degree 1 over the finite field \( \mathbb{F}_2 \), Razborov and Smolensky showed that over other fields (such as \( \mathbb{F}_3 \)), any degree-\( o(\sqrt{n}) \) polynomial disagrees with Parity on a large fraction of inputs—and that is enough to imply that \( \text{Parity} \notin \text{AC}^0 \). In other words, we simply need to find a type of polynomial approximation that works for \( \text{AC}^0 \) circuits, but does not work for the Fourier Checking problem. If true, Conjecture 28 (the Low-Fat Sandwich Conjecture) provides exactly such a type of approximation.

But this raises another question: what is the significance of the “low-fat” requirement in Conjecture 28? Why, of all things, do we want our approximating polynomial \( p \) to be expressible as a linear combination of terms, \( p(x) = \sum_C \alpha_C C(x) \), such that \( \sum_C |\alpha_C|^2 \leq n^{o(1)} \)?

The answer takes us to the heart of what an oracle separation between \( \text{BQP} \) and \( \text{PH} \) would have to accomplish. Notice that, although the polynomial \( p \) from equation (2) solved the Fourier Checking problem, it did so only by cancelling massive numbers of positive and negative terms, then representing the answer by the tiny residue left over. Not coincidentally, this sort of cancellation is a central feature of quantum algorithms. By contrast, Theorem 29 essentially says that, if a polynomial \( p \) does not involve such massive cancellations, but is instead more “conservative” and “reasonable” (like the polynomials that arise from classical decision trees), then \( p \) cannot distinguish almost \( k \)-wise independent distributions from the uniform distribution, and therefore cannot solve Fourier Checking. If Conjecture 28 holds, then every small-depth circuit can be approximated, not just by any low-degree polynomial, but by a “conservative,” “reasonable” low-degree polynomial—one with a bound on the coefficients that prevents massive cancellations. This would prove that Fourier Checking has no small constant-depth circuits, and hence that there exists an oracle separating \( \text{BQP} \) from \( \text{PH} \).

This brings us to the fourth and final question: how might one prove Conjecture 28? In particular, is it possible that some trivial modification of Braverman’s proof [15] would give low-fat sandwiching polynomials, thereby establishing the GLN Conjecture?

While we cannot rule this out, we believe that the answer is no. For examining Braverman’s proof, we find that it combines two kinds of polynomial approximations of \( \text{AC}^0 \) circuits: that of Linial-Mansour-Nisan [30], and that of Razborov [32] and Smolensky [39]. Unfortunately, neither LMN nor Razborov-Smolensky gives anything like the control over the approximating polynomial’s coefficients that Conjecture 28 demands. LMN simply takes the Fourier transform of an \( \text{AC}^0 \) function and deletes the high-order coefficients; while Razborov-Smolensky approximates each OR gate by a product of randomly-chosen linear functions. Both techniques produce approximating polynomials with a huge number of monomials, and no reasonable bound on their coefficients. While it is conceivable that those polynomials satisfy the low-fat condition anyway—because of some non-obvious representation as a linear combination of terms—certainly neither LMN nor Razborov-Smolensky gives any idea what that representation would look like. Thus, we suspect that, to get the desired control over the coefficients, one will need more “constructive” proofs of both the LMN and Razborov-Smolensky theorems. Such proofs would likely be of great interest to circuit complexity and computational learning theory for independent reasons.
8 Open Problems

First, of course, prove the GLN Conjecture, or prove the existence of an oracle \( A \) relative to which \( \text{BQP}^A \not\subset \text{PH}^A \) by some other means. A natural first step would be to prove the GLN Conjecture for the special case of DNFs: as shown in Theorem 26, this would imply an oracle \( A \) relative to which \( \text{BQP}^A \not\subset \text{AM}^A \). We have offered a $200 prize for the \text{PH} case and a $100 prize for the \text{AM} case.\(^{16}\)

Second, it would be of interest to prove the GLN Conjecture for classes of functions weaker than (or incomparable with) DNFs: for example, monotone DNFs, read-once formulas, and read-\( k \)-times formulas.

Third, can we give an example of a Boolean function \( f : \{0,1\}^n \to \{-1,1\} \) that is well-approximated by a low-degree polynomial, but not by a low-degree low-fat polynomial? Here is a more concrete version of the challenge: let

\[
\|f - p\| := \mathbb{E}_{x \in \{0,1\}^n} \left[ (f(x) - p(x))^2 \right].
\]

Then find a Boolean function \( f \) for which

(i) there exists a degree-\( n^{o(1)} \) polynomial \( p : \mathbb{R}^n \to \mathbb{R} \) such that \( \|f - p\| = o(1) \), but

(ii) there does not exist a degree-\( n^{o(1)} \) polynomial \( q : \mathbb{R}^n \to \mathbb{R} \) such that \( \|f - q\| = o(1) \) and \( q \) can be written as a linear combination of terms, \( q(x) = \sum_C \alpha_C C(x) \), with \( \sum_C |\alpha_C|^2 2^{-|C|} = n^{o(1)} \).

Fourth, can we give an oracle relative to which \( \text{BQP} \not\subset \text{IP} \)? What about an oracle relative to which \( \text{BQP} \not\equiv \text{IP} \)?

Fifth, what other implications does the GLN Conjecture have? If we assume it, can we address other longstanding open questions in quantum complexity theory, such as those discussed in Section 1.1? For example, can we give an oracle relative to which \( \text{NP} \subseteq \text{BQP} \) but \( \text{PH} \not\subset \text{BQP} \), or an oracle relative to which \( \text{NP} \subseteq \text{BQP} \) and \( \text{PH} \) is infinite?

Sixth, how much can we say about the \( \text{BQP} \) versus \( \text{PH} \) question in the unrelativized world? As one concrete challenge, can we find a nontrivial way to “realize” the Fourier Checking oracle (in other words, an explicit computational problem that is solvable using Fourier Checking)?

Seventh, how far can the gap between the success probabilities of \( \text{FBQP} \) and \( \text{FBPP}^{\text{PH}} \) algorithms be improved? Theorem 15 gave a relation for which a quantum algorithm succeeds with probability \( 1 - c^{-n} \), whereas any \( \text{FBPP}^{\text{PH}} \) algorithm succeeds with probability at most 0.99. By changing the success criterion for Fourier Fishing—basically, by requiring the classical algorithm to output \( z_1, \ldots, z_n \) such that \( \hat{f}_1(z_1)^2, \ldots, \hat{f}_n(z_n)^2 \) are distributed “almost exactly as they would be in the quantum algorithm”—one can improve the 0.99 to \( 1/2 + \epsilon \) for any \( \epsilon > 0 \). However, improving the constant further might require a direct product theorem for \( \text{AC}^0 \) circuits solving Fourier Fishing.

Eighth, we mentioned before that Fourier Checking yields the largest gap between randomized and quantum query complexities yet known: namely, \( \text{X}^{\Omega(1)} \) versus \( O(1) \). By generalizing

\(^{16}\)See http://scottaaronson.com/blog/?p=381

\(^{17}\)If we let the verifier transmit unentangled qubits to the prover, then the resulting class \( \text{IP}_{\text{BQP}}^{(\theta)} \) actually equals \( \text{BQP} \), as recently shown by Broadbent, Fitzsimons, and Kashefi [16] (see also Aharonov, Ben-Or, and Eban [5]). It is not known whether this \( \text{IP}_{\text{BQP}}^{(\theta)} = \text{BQP} \) result relativizes; we conjecture that it does not.
Fourier Checking to involve $k$ Boolean functions rather than just two, we conjecture that it should be possible to get a decision problem whose bounded-error randomized query complexity is $\Omega \left( N^{1-1/k} \right)$ and whose quantum query complexity is only $O(k)$, for any $k$. Perhaps this gap can be improved even further?

9 Acknowledgments

I thank Louay Bazzi for reformulating the GLN Conjecture as the Low-Fat Sandwich Conjecture, Andy Drucker, Lance Fortnow, and Sasha Razborov for helpful discussions, and an anonymous reviewer for comments.

References


