Electronic Colloquium on Computational Complexity, Report No. 155 (2014)

Forrelation: A Problem that Optimally Separates Quantum from Classical Computing

Scott Aaronson[*]
MIT

Andris Ambainis[†]
University of Latvia

Abstract

We achieve essentially the largest possible separation between quantum and classical query complexities. We do so using a property-testing problem called FORRELATION, where one needs to decide whether one Boolean function is highly correlated with the Fourier transform of a second function. This problem can be solved using 1 quantum query, yet we show that any randomized algorithm needs $\Omega(\sqrt{N}/\log N)$ queries (improving an $\Omega(N^{1/4})$ lower bound of Aaronson). Conversely, we show that this 1 versus $\widetilde{\Omega}(\sqrt{N})$ separation is optimal: indeed, any t-query quantum algorithm whatsoever can be simulated by an $O\left(N^{1-1/2t}\right)$-query randomized algorithm. Thus, resolving an open question of Buhrman et al. from 2002, there is no partial Boolean function whose quantum query complexity is constant and whose randomized query complexity is linear. We conjecture that a natural generalization of FORRELATION achieves the optimal t versus $\Omega\left(N^{1-1/2t}\right)$ separation for all t. As a bonus, we show that this generalization is BQP-complete. This yields what's arguably the simplest BQP-complete problem yet known, and gives a second sense in which FORRELATION "captures the maximum power of quantum computation."

1 Introduction

Since the work of Simon [23] and Shor [22] two decades ago, we have had powerful evidence that quantum computers can achieve exponential speedups over classical computers. Of course, for problems like FACTORING, these speedups are conjectural at present: we cannot rule out that a fast classical factoring algorithm might exist. But in the *black-box model*, which captures most known quantum algorithms, exponential and even larger speedups can be *proved*. We know, for example, that PERIOD-FINDING (a natural abstraction of the problem solved by Shor's algorithm) is solvable with only $O(1)$ quantum queries, but requires $N^{\Omega(1)}$ classical randomized queries, where N is the number of input elements [12, 10, 17]. We also know that SIMON'S PROBLEM is solvable with $O(\log N)$ quantum queries, but requires $\Omega(\sqrt{N})$ classical queries; and that a similar separation holds for the GLUED-TREES problem introduced by Childs et al. [11, 16].[1]

To us, these results raise an extremely interesting question:

[*] Email: aaronson@csail.mit.edu. Supported by an NSF Waterman Award.

[†] The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 600700 (QALGO) and ERC Advanced Grant MQC.

[1] However, in all these cases the queries are non-Boolean. If we insist on Boolean queries, then the quantum query complexities get multiplied by an $O(\log N)$ factor.

ISSN 1433-8092

- **"The Speedup Question."** *Within the black-box model, just how large of a quantum speedup is possible? For example, could there be a function of N bits with a quantum query complexity of* 1*, but a classical randomized query complexity of $\Omega(N)$?*

One may object: once we know that exponential and even larger quantum speedups are possible in the black-box model, who cares about the exact limit? In our view, the central reason to study the Speedup Question is that doing so can help us better understand the nature of quantum speedups themselves. For example, can all exponential quantum speedups be seen as originating from a common cause? Is there a single problem or technique that captures the advantages of quantum over classical query complexity, in much the same way that random sampling could be said to capture the advantages of randomized over deterministic query complexity?

As far as we know, the Speedup Question was first posed by Buhrman et al. [8] around 2002, in their study of quantum property-testing. Specifically, Buhrman et al. asked whether there is any property of N-bit strings that exhibits a "maximal" separation: that is, one that requires $\Omega(N)$ queries to test classically, but only $O(1)$ quantumly. The best separation they could find, based on Simon's problem, was "deficient" on both ends: it required $\Omega(\sqrt{N})$ queries to test classically, and $O(\log N \log\log N)$ quantumly.

Since then, there has been only sporadic progress on the Speedup Question. In 2009, Aaronson [1] introduced the FORRELATION problem—a problem that we will revisit in this paper—and showed that it was solvable with only 1 quantum query, but required $\Omega(N^{1/4})$ classical randomized queries. In 2010, Chakraborty et al. [10] argued that PERIOD-FINDING gives a different example of an $O(1)$ versus $\widetilde{\Omega}(N^{1/4})$ quantum/classical gap; there, however, we only get an $O(1)$-query quantum algorithm if we allow non-Boolean queries.

Earlier, in 2001, de Beaudrap, Cleve, and Watrous [6] had given what they described as a black-box problem that was solvable with 1 quantum query, but that required $\Omega(N^{1/4})$ or $\Omega(\sqrt{N})$ classical randomized queries (depending on how one defines the "input size" N). However, de Beaudrap et al. were not working within the usual model of quantum query complexity. Normally, one provides "black-box access" to a function f, meaning that the quantum algorithm can apply a unitary transformation that maps basis states of the form $|x, y\rangle$ to basis states of the form $|x, y \oplus f(x)\rangle$ (or $|x\rangle$ to $(-1)^{f(x)}|x\rangle$, if f is Boolean). By contrast, for their separation, de Beaudrap et al. had to assume the ability to map basis states of the form $|x, y\rangle$ to basis states of the form $|x, \pi(y + sx)\rangle$, for some unknown permutation π and hidden shift s.

1.1 Our Results

This paper has two main contributions—the largest quantum black-box speedup yet known, and a proof that that speedup is essentially optimal—as well as many smaller related contributions.

1.1.1 Maximal Quantum/Classical Separation

In Section 4, we undertake a detailed study of the FORRELATION problem, which Aaronson [1] introduced for a different purpose than the one that concerns us here (he was interested in an oracle separation between BQP and the polynomial hierarchy).[2] In FORRELATION, we are given access to two Boolean functions $f, g : \{0,1\}^n \to \{-1, 1\}$. We want to estimate the amount of

[2] Also, in [1], the problem was called "Fourier Checking."

correlation between f and the Fourier transform of g—that is, the quantity

$$\Phi_{f,g} := \frac{1}{2^{3n/2}} \sum_{x,y \in \{0,1\}^n} f(x) (-1)^{x \cdot y} g(y).$$

It is not hard to see that $|\Phi_{f,g}| \leq 1$ for all f, g. The problem is to decide, say, whether $|\Phi_{f,g}| \leq \frac{1}{100}$ or $\Phi_{f,g} \geq \frac{3}{5}$, promised that one of these is the case.[3] Here and throughout this paper, the "input size" is taken to be $N := 2^n$.

One can give (see Section 3) a quantum algorithm that solves Forrelation, with bounded probability of error, using only 1 quantum query. Intuitively, however, the property of f and g being "forrelated" (that is, having large $\Phi_{f,g}$ value) is an extremely global property, which should not be apparent to a classical algorithm until it has queried a significant fraction of the entire truth tables of f and g. And indeed, improving an $\Omega\left(N^{1/4}\right)$ lower bound of Aaronson [1], in Section 4 we show the following:

Theorem 1 *Any classical randomized algorithm for* Forrelation *must make* $\Omega(\frac{\sqrt{N}}{\log N})$ *queries.*

Theorem 1 yields *the largest quantum versus classical separation yet known* in the black-box model. As we will show in Appendix 10, Theorem 1 also implies the largest *property-testing* separation yet known—for with some work, one can recast Forrelation (or rather, its negation) as a property that is testable with only 1 query quantumly, but that requires $\Omega(\frac{\sqrt{N}}{\log N})$ queries to test classically.

We deduce Theorem 1 as a consequence of a more general result: namely, a lower bound on the classical query complexity of a problem called Gaussian Distinguishing. Here we are given oracle access to a collection of $\mathcal{N}(0,1)$ real Gaussian random variables, $x_1, \ldots, x_M$. We are asked to decide whether the variables are all independent, or alternatively, whether they lie in a known low-dimensional subspace of $\mathbb{R}^M$: one that induces a covariance of at most ε between each pair of variables, while keeping each variable an $\mathcal{N}(0,1)$ Gaussian individually. We show the following:

Theorem 2 Gaussian Distinguishing *requires* $\Omega\left(\frac{1/\varepsilon}{\log(M/\varepsilon)}\right)$ *classical randomized queries.*

Theorem 1 is then simply a (discretized) special case of Theorem 2, with $M = 2N$ and $\varepsilon = 1/\sqrt{N}$. Beyond that, it seems to us that Theorem 2 could have independent applications in statistics and machine learning.

1.1.2 Proof of Optimality

In Section 5, we show that the quantum/classical query complexity separation achieved by the Forrelation problem is close to the best possible. More generally:

[3] The reason for the asymmetry—i.e., for promising that $\Phi_{f,g}$ is positive if its absolute value is large, but not if its absolute value is small—is a bit technical. On the one hand, we want the "unforrelated" case to encompass almost all randomly-chosen functions f, g. On the other hand, we also want Forrelation to be solvable using only 1 quantum query. If we had promised $|\Phi_{f,g}| \geq \frac{3}{5}$, rather than $\Phi_{f,g} \geq \frac{3}{5}$, then we would only know a 2-query quantum algorithm. In any case, none of these choices make a big difference to our results.

Theorem 3 *Let Q be any quantum algorithm that makes $t = O(1)$ queries to an N-bit string $X \in \{0,1\}^N$. Then we can estimate $\Pr[Q \text{ accepts } X]$, to constant additive error and with high probability, by making only $O(N^{1-1/2t})$ classical randomized queries to X.*[4] *Moreover, the randomized queries are nonadaptive.*

So for example, every 1-query quantum algorithm can be simulated by an $O(\sqrt{N})$-query classical randomized algorithm, every 2-query quantum algorithm can be simulated by an $O(N^{3/4})$-query randomized algorithm, and so on. Theorem 3 resolves the open problem of Buhrman et al. [8] in the negative: it shows that there is no problem (property-testing or otherwise) with a constant versus linear quantum/classical query complexity gap. Theorem 3 does not rule out the possibility of an $O(\log N)$ versus $\widetilde{\Omega}(N)$ gap, and indeed, we conjecture that such a gap is possible.

Once again, we deduce Theorem 3 as a consequence of a more general result, which might have independent applications to classical sublinear algorithms. Namely:

Theorem 4 *Every degree-k real polynomial $p : \{-1,1\}^N \to \mathbb{R}$ that is*

(i) bounded in $[-1,1]$ at every Boolean point, and

(ii) "block-multilinear" (that is, the variables can be partitioned into k blocks, such that every monomial is the product of one variable from each block),

can be approximated to within $\pm\varepsilon$, with high probability, by nonadaptively querying only $O((N/\varepsilon^2)^{1-1/k})$ of the variables.

In the statement of Theorem 4, we strongly conjecture that condition (ii) can be removed. If so, then we would obtain a sublinear algorithm to estimate any bounded, constant-degree real polynomial. In Appendix 8, we show that condition (ii) can indeed be removed in the special case $k = 2$.

1.1.3 k-fold Forrelation

In Section 6, we study a natural generalization of Forrelation. In k-fold Forrelation, we are given access to k Boolean functions $f_1, \ldots, f_k : \{0,1\}^n \to \{-1,1\}$. We want to estimate the "twisted sum"

$$\Phi_{f_1,\ldots,f_k} := \frac{1}{2^{(k+1)n/2}} \sum_{x_1,\ldots,x_k \in \{0,1\}^n} f_1(x_1)(-1)^{x_1 \cdot x_2} f_2(x_2)(-1)^{x_2 \cdot x_3} \cdots (-1)^{x_{k-1} \cdot x_k} f_k(x_k).$$

It is not hard to show that $|\Phi_{f_1,\ldots,f_k}| \le 1$ for all $f_1, \ldots, f_k$. The problem is to decide, say, whether $|\Phi_{f_1,\ldots,f_k}| \le \frac{1}{100}$ or $\Phi_{f_1,\ldots,f_k} \ge \frac{3}{5}$, promised that one of these is the case.

One can give (see Section 3) a quantum algorithm that solves k-fold Forrelation, with bounded error probability, using only $\lceil k/2 \rceil$ quantum queries. In Section 6, we show, conversely, that k-fold Forrelation "captures the full power of quantum computation":

Theorem 5 *If $f_1, \ldots, f_k$ are described explicitly (say, by circuits to compute them), and $k = \operatorname{poly}(n)$, then k-fold* Forrelation *is a* BQP*-complete promise problem.*

[4] The reason for the condition $t = O(1)$ is that, in the bound $O(N^{1-1/2t})$, the big-O hides a multiplicative factor of $\exp(t)$. Thus, we can obtain a nontrivial upper bound on query complexity as long as $t = o(\sqrt{\log N})$.

We do not know of any complete problem for quantum computation that is more self-contained than this. Not only can one state the k-fold FORRELATION problem without any notions from quantum mechanics, one does not need any nontrivial mathematical notions, like the condition number of a matrix or the Jones polynomial of a knot.

We conjecture, moreover, that k-fold FORRELATION achieves the optimal $k/2$ versus $\widetilde{\Omega}(N^{1-1/k})$ quantum/classical query complexity separation for all even k. If so, then there are *two* senses in which k-fold FORRELATION captures the power of quantum computation.

1.1.4 Other Results

The paper also includes several other results.

In Appendix 7, we study the largest possible quantum/classical separations that are achievable for *approximate sampling* and *relation* problems. We show that there exists a sampling problem—namely, FOURIER SAMPLING of a Boolean function—that is solvable with 1 quantum query, but requires $\Omega(N/\log N)$ classical queries. By our previous results, this exceeds the largest quantum/classical gap that is possible for decision problems.

In Appendix 8, we generalize our result that every 1-query quantum algorithm can be simulated using $O(\sqrt{N})$ randomized queries, to show that *every bounded degree-*2 *real polynomial* $p : \{-1,1\}^N \to [-1,1]$ can be estimated using $O(\sqrt{N})$ randomized queries. We conjecture that this can be generalized, to show that every bounded degree-k real polynomial can be estimated using $O(N^{1-1/k})$ randomized queries.

In Appendix 9, we extend our $\Omega(\frac{\sqrt{N}}{\log N})$ randomized lower bound for the FORRELATION problem, to show a $\Omega(\frac{\sqrt{N}}{\log^{7/2} N})$ lower bound for k-fold FORRELATION for any $k \geq 2$. We conjecture that the right lower bound is $\widetilde{\Omega}(N^{1-1/k})$, but even generalizing our $\widetilde{\Omega}(\sqrt{N})$ lower bound to the k-fold case is nontrivial.

1.2 Techniques

1.2.1 Randomized Lower Bound

Proving that any randomized algorithm for FORRELATION requires $\Omega(\frac{\sqrt{N}}{\log N})$ queries is surprisingly involved. As we mentioned in Section 1.1.1, the first step, following the work of Aaronson [1], is to convert FORRELATION into an analogous problem involving real Gaussian variables. In REAL FORRELATION, we are given oracle access to two real functions $f, g : \{0,1\}^n \to \mathbb{R}$, and are promised either that (i) every $f(x)$ and $g(y)$ value is an independent $\mathcal{N}(0,1)$ Gaussian, or else (ii) every $f(x)$ value is an independent $\mathcal{N}(0,1)$ Gaussian, while every $g(y)$ value equals $\hat{f}(y)$ (i.e., the Fourier transform of f evaluated at y). The problem is to decide which holds. Using a rounding reduction, we show that any query complexity lower bound for REAL FORRELATION implies the same lower bound for FORRELATION itself.

Making the problem continuous allows us to adopt a geometric perspective. In this perspective, we are given oracle access to a real vector $v \in \mathbb{R}^{2N}$, whose $2N$ coordinates consist of all values $f(x)$ and all values $g(y)$ (recall that $N = 2^n$). We are trying to distinguish the case where v is simply an $\mathcal{N}(0,1)^{2N}$ Gaussian, from the case where v is confined to an N-dimensional subspace of $\mathbb{R}^{2N}$—namely, the subspace defined by $g = \hat{f}$. Now, suppose that values $f(x_1), \ldots, f(x_t)$ and $g(y_1), \ldots, g(y_u)$ have already been queried. Then we can straightforwardly calculate the Bayesian posterior probabilities for being in case (i) or case (ii). For case (i), the probability turns out to depend solely on the squared 2-norm of the vector of empirical data seen so far:

$$\Pr\left[\text{case (i)}\right] \propto \exp\left(-\frac{\Delta_{\mathrm{i}}}{2}\right),$$

where

$$\Delta_{\mathrm{i}} = f\left(x_1\right)^2 + \cdots + f\left(x_t\right)^2 + g\left(y_1\right)^2 + \cdots + g\left(y_u\right)^2.$$

For case (ii), by contrast, the probability is proportional to $\exp(-\Delta_{\mathrm{ii}}/2)$, where Δ_{ii} is the minimum squared 2-norm of any point $f \in \mathbb{R}^N$ compatible with all the data seen so far, as well as with the linear constraint $g = \hat{f}$. Let $\mathcal{V} = \left\{ |1\rangle, \ldots, |N\rangle, |\hat{1}\rangle, \ldots, |\hat{N}\rangle \right\}$ be the set of $2N$ unit vectors in $\mathbb{R}^N$ that consists of all N elements of the standard basis, together with all N elements of the Fourier basis. Then Δ_{ii}, in turn, can be calculated using a process of Gram-Schmidt orthogonalization, on the vectors in $\mathcal{V}$ corresponding to the f-values and g-values that have been queried so far.

Our goal is to show that, with high probability, Δ_{i} and Δ_{ii} *remain close to each other*, even after a large number of queries have been made—meaning that the algorithm has not yet succeeded in distinguishing case (i) from case (ii) with non-negligible bias. To show this, the central fact we rely on is that the vectors in $\mathcal{V}$ are *nearly-orthogonal*: that is, for all $|v\rangle, |w\rangle \in \mathcal{V}$, we have $|\langle v|w\rangle| \leq \frac{1}{\sqrt{N}}$. Intuitively, this means that, if we restrict attention to any small subset S of f-values and g-values, then while correlations exist among those values, the correlations are *weak*: "to a first approximation," we have simply asked for the projections of a Gaussian vector onto $|S|$ orthogonal directions, and have therefore received $|S|$ uncorrelated answers.

From this perspective, the key question is: how many values can we query until the "orthogonal approximation" breaks down (meaning that we notice the correlations)? In his previous work, Aaronson [1] showed that the approximation holds until $\Omega\left(N^{1/4}\right)$ queries are made. Indeed, he proved a stronger statement: even if the x's and y's are chosen *nondeterministically*, still $\Omega\left(N^{1/4}\right)$ values must be revealed until we have a *certificate* showing that we are in case (i) or case (ii) with high probability.

To improve the lower bound from $\Omega\left(N^{1/4}\right)$ to the optimal $\widetilde{\Omega}(\sqrt{N})$, there are several hurdles to overcome.

Aaronson had assumed, conservatively, that the deviations from orthogonality all "pull in the same direction." As a first step, we notice instead that the deviations follow an unbiased random walk, with some positive and others negative—the martingale property arising from the fact that the algorithm can control which x's and y's to query, but not the values of $f\left(x\right)$ and $g\left(y\right)$. We then use a Gaussian generalization of Azuma's inequality to upper-bound the sum of the deviations. Doing this improves the lower bound from $\Omega\left(N^{1/4}\right)$ to $\widetilde{\Omega}\left(N^{1/3}\right)$, but we then hit an apparent barrier.

In this work, we explain the $\Omega\left(N^{1/3}\right)$ barrier, by exhibiting a "model problem" that is extremely similar to Real Forrelation (in particular, has exactly the same near-orthogonality property), yet is solvable with only $O\left(N^{1/3}\right)$ queries, by exploiting adaptivity. However, we then break the barrier, by using the fact that, for Real Forrelation (but *not* for the model problem), the total number of vectors in $\mathcal{V}$ is only $N^{O(1)}$. This fact lets us use the Gaussian Azuma's inequality a second time, to upper-bound not only the *sum* of all the deviations from orthogonality, but the individual deviations themselves. Implementing this yields a lower bound of $\widetilde{\Omega}\left(N^{2/5}\right)$: better than $\widetilde{\Omega}\left(N^{1/3}\right)$, but still not all the way up to $\widetilde{\Omega}(\sqrt{N})$. However, we then notice that we can apply Azuma's inequality *recursively*—once for each layer of the Gram-Schmidt orthogonalization process—to get better and better upper bounds on the deviations from orthogonality. Doing so gives us a sequence of lower bounds $\widetilde{\Omega}\left(N^{3/7}\right)$, $\widetilde{\Omega}\left(N^{4/9}\right)$, $\widetilde{\Omega}\left(N^{5/11}\right)$, etc., with the ultimate limit of the process being $\Omega(\frac{\sqrt{N}}{\log N})$.

1.2.2 Randomized Upper Bound

Why did we have to work so hard to prove a $\widetilde{\Omega}(\sqrt{N})$ lower bound on the randomized query complexity of FORRELATION? Our other main result provides one possible explanation: namely, we are here scraping up against the "ceiling" of the possible separations between randomized and quantum query complexity. In particular, *any* quantum algorithm that makes 1 query to a Boolean input $X \in \{0,1\}^N$, can be simulated by a randomized algorithm (in fact, a nonadaptive randomized algorithm) that makes $O(\sqrt{N})$ queries to X. More generally, any quantum algorithm that makes $t = O(1)$ queries to X, can be simulated by a nonadaptive randomized algorithm that makes $O(N^{1-1/2t})$ queries to X.

The proof of this result consists of three steps. The first involves the simulation of quantum algorithms by low-degree polynomials. In 1998, Beals et al. [5] famously observed that, if a quantum algorithm makes t queries to a Boolean input $X \in \{-1,1\}^N$, then $p(X)$, the probability that the algorithm accepts X, can be written as a multilinear polynomial in X of degree at most $2t$. We extend this result of Beals et al., in a way that might be of independent interest for quantum lower bounds. Namely, we observe that every t-query quantum algorithm gives rise, not merely to a multilinear polynomial, but to a *block-multilinear* polynomial. By this, we mean a degree-$2t$ polynomial q that takes as input $2t$ blocks of N variables each, and whose every monomial contains exactly one variable from each block. If we repeat the input $X \in \{-1,1\}^N$ across all $2t$ blocks, then $q(X,\ldots,X)$ represents the quantum algorithm's acceptance probability on X. However, the key point is that $q(Y)$ is bounded in $[-1,1]$ for *any* Boolean input $Y \in \{-1,1\}^{2tN}$.

This leads to a new complexity measure for Boolean functions f: the *block-multilinear approximate degree* $\widetilde{\mathrm{bmdeg}}(f)$, which lower-bounds the quantum query complexity $\mathrm{Q}(f)$ just as $\widetilde{\deg}(f)$ does, but which might provide a tighter lower bound in some cases. (Indeed, we do not even know whether there is any asymptotic separation between $\mathrm{Q}(f)$ and $\widetilde{\mathrm{bmdeg}}(f)$, whereas Ambainis [4] showed an asymptotic separation between $\mathrm{Q}(f)$ and $\widetilde{\deg}(f)$.)

Once we have our quantum algorithm's acceptance probability in the form of a block-multilinear polynomial q, the second step is to *preprocess* q, to make it easier to estimate using random sampling. The basic problem is that q might be highly "unbalanced": certain variables might be hugely influential. Such variables are essential to query, but examining the form of q does not make it obvious which variables these are. To deal with this, we repeatedly perform an operation called "variable-splitting," which consists of identifying an influential variable x_i, then replacing every occurrence of x_i in q by $\frac{1}{m}(x_{i,1} + \cdots + x_{i,m})$, where $x_{i,1},\ldots,x_{i,m}$ are newly-created variables set equal to x_i. The point of doing this is that each $x_{i,j}$ will be less influential in q than x_i itself was, thereby yielding a more balanced polynomial. We show that variable-splitting can achieve the desired balance by introducing at most $\exp(t) \cdot O(N)$ new variables, which is linear in N for constant t.

Once we have a balanced polynomial q, the last step is to give a query-efficient randomized algorithm to estimate its value. Our algorithm is the simplest one imaginable: we simply choose $O(N^{1-1/2t})$ variables uniformly at random, query them, then form an estimate $\widetilde{q}$ of q by summing only those monomials all of whose variables were queried. The hard part is to prove that this estimator *works*—i.e., that its variance is bounded. The proof of this makes heavy use of the balancedness property that was ensured by the preprocessing step.

Examining our estimation algorithm, an obvious question is whether it was essential that q be block-multilinear, or whether the algorithm could be extended to *all* bounded low-degree polynomials. In Appendix 8, we take a first step toward answering that question, by giving an $O(\sqrt{N})$-query randomized algorithm to estimate any bounded degree-2 polynomial in N Boolean variables. Once we drop block-multilinearity, our variable-splitting procedure no longer works, so we rely instead on Fourier-analytic results of Dinur et al. [14] to identify influential variables which we then split.

1.2.3 Other Results

BQP-Completeness. The proof that the k-fold Forrelation problem is PromiseBQP-complete is simple, once one has the main idea. The sum that defines k-fold Forrelation is, itself, an output amplitude for a particular kind of quantum circuit, which consists entirely of Hadamard and f-phase gates (i.e., gates that map $|x\rangle$ to $(-1)^{f(x)}|x\rangle$ for some Boolean function f). Since the Hadamard and CCPHASE gates (corresponding to $f(x,y,z)=xyz$) are known to be universal for quantum computation, one might think that our work is done. The difficulty is that the quantum circuit for k-fold Forrelation contains a Hadamard gate on every qubit, between every pair of f-phase gates, *whether we wanted Hadamards there or not.* Thus, if we want to encode an arbitrary quantum circuit, then we need some way of *canceling* unwanted Hadamards, while leaving the wanted ones. We achieve this via a gadget construction.

Separation for Sampling Problems. To achieve a 1 versus $\Omega(N/\log N)$ quantum/classical query complexity separation for a sampling problem, we consider Fourier Sampling: the problem, given oracle access to a Boolean function $f:\{0,1\}^n \to \{-1,1\}$, of outputting a string $y\in\{0,1\}^n$ with probability approximately equal to $\hat{f}(y)^2$. This problem is trivially solvable with 1 quantum query, but proving a $\Omega(N/\log N)$ classical lower bound takes a few pages of work. The basic idea is to concentrate on the probability of a *single* string—say, $y=0^n$—being output. Using a binomial calculation, we show that this probability cannot depend on f's truth table in the appropriate way unless $\Omega(N/\log N)$ function values are queried.

Lower Bound for k-Fold Forrelation. Once we have a $\Omega(\frac{\sqrt{N}}{\log N})$ randomized lower bound for Forrelation, one might think it would be trivial to prove the same lower bound for k-fold Forrelation: just reduce one to the other! However, Forrelation does not embed in any clear way as a subproblem of k-fold Forrelation. On the other hand, given an instance of k-fold Forrelation, suppose we "give away for free" the complete truth tables of all but two of the functions. In that case, we show that the induced subproblem on the remaining two functions is an instance of Gaussian Distinguishing to which, with high probability, our lower bound techniques can be applied. Pursuing this idea leads to our $\Omega(\frac{\sqrt{N}}{\log^{7/2} N})$ lower bound on the randomized query complexity of k-fold Forrelation, for all $k \geq 2$.

Property-Testing Separation. To turn our quantum versus classical separation for the Forrelation problem into a property-testing separation, we need to prove two interesting statements. The first is that function pairs $\langle f,g\rangle$ that are far in Hamming distance from the set of all pairs with low forrelation, actually have high forrelation. The second is that "generic" function pairs $\langle f,g\rangle$ and $\langle f',g'\rangle$ that have small Hamming distance from one another, are close in their forrelation values as well. In fact, we will prove both of these statements for the general case of k-fold Forrelation.

1.3 Discussion

To summarize, this paper proves the largest separation between classical and quantum query complexities yet known, and it also proves that that separation is in some sense optimal. These results put us in a position to pose an intriguing open question:

> Among all the problems that admit a superpolynomial quantum speedup, is there any whose classical randomized query complexity is $\gg \sqrt{N}$?

Strikingly, if we look at the known problems with superpolynomial quantum speedups, for every one of them the classical randomized lower bound seems to hit a "ceiling" at $\sqrt{N}$. Thus, Simon's Problem has quantum query complexity $O(\log N)$ and randomized query complexity $\widetilde{\Theta}(\sqrt{N})$; the Glued-Trees problem of Childs et al. [11] has quantum query complexity $\log^{O(1)}(N)$ and randomized query complexity $\widetilde{\Theta}(\sqrt{N})$;[5] and Forrelation has quantum query complexity 1 and randomized query complexity $\widetilde{\Theta}(\sqrt{N})$.

If we insist on making the randomized query complexity $\Omega(N^{1/2+c})$, for some $c > 0$, and then try to minimize the quantum query complexity, then the best thing we know how to do is to take the OR of N^{2c} independent instances of Forrelation, each of size N^{1-2c}. This gives us a problem whose quantum query complexity is $\Theta(N^c)$,[6] and whose classical randomized query complexity is $\widetilde{\Theta}(N^{1/2+c})$.[7] Of course, this is not an exponential separation.

In this paper, we gave a candidate for a problem that breaks the "$\sqrt{N}$ barrier": namely, k-fold Forrelation. Indeed, we conjecture that k-fold Forrelation achieves the *optimal* separation for all $k = O(1)$, requiring $\widetilde{\Omega}\left(N^{1-1/k}\right)$ classical randomized queries but only $\lceil k/2 \rceil$ quantum queries.[8] Proving this conjecture is an enticing problem. Unfortunately, k-fold Forrelation becomes extremely hard to analyze when $k > 2$, because we can no longer view the functions $f_1, \ldots, f_k$ as confined to a low-dimensional subspace: now we have to view them as confined to a low-dimensional *manifold*, which is defined by degree-$(k-1)$ polynomials. As such, we can no longer compute posterior probabilities by simply appealing to the rotational invariance of the Gaussian measure, which made our lives easier in the $k = 2$ case. Instead we need to calculate integrals over a nonlinear manifold.

Short of proving our conjecture about k-fold Forrelation, it would of course be nice to find *any* partial Boolean function whose quantum query complexity is polylog N, and whose randomized query complexity is $N^{1/2+\Omega(1)}$.

[5] The randomized lower bound for Glued-Trees proved by Childs et al. [11] was only $\Omega(N^{1/6})$. However, Fenner and Zhang [16] improved the lower bound to $\Omega(N^{1/3})$; and if we allow a success probability that is merely (say) 1/3, rather than exponentially small, then their bound can be improved further, to $\Omega(\sqrt{N})$. In the other direction, we are indebted to Shalev Ben-David for proving that Glued-Trees can be solved deterministically using only $O(\sqrt{N} \log N)$ queries (or $O(\sqrt{N} \log^2 N)$, if the queries are required to be Boolean). For his proof, see http://cstheory.stackexchange.com/questions/25279/the-randomized-query-complexity-of-the-conjoined-trees-problem

[6] Here the upper bound comes from combining Grover's algorithm with the Forrelation algorithm: the "naïve" way of doing this would produce an additional $\log N$ factor for error reduction, but it is well-known that that log factor can be eliminated [19]. Meanwhile, the lower bound comes from the optimality of Grover's algorithm.

[7] Here the upper bound comes from simply taking the best randomized Forrelation algorithm, which uses $O(\sqrt{N^{1-2c}})$ queries, and running it N^{2c} times, with an additional $\log N$ factor for error reduction. Meanwhile, the lower bound comes from combining this paper's $\Omega(\sqrt{N}/\log N)$ lower bound for Forrelation, with a general result stating that the randomized query complexity of $\mathrm{OR}(f, \ldots, f)$, the OR of k disjoint copies of a function f, is $\Omega(k)$ times the query complexity of a single copy. This result can be proved by adapting ideas from a direct product theorem for randomized query complexity given by Drucker [15] (we thank A. Drucker, personal communication).

[8] And perhaps k-fold Forrelation continues to give optimal separations, all the way up to $k = O(\log N)$.

Another problem we leave is to generalize our $O\left(N^{1-1/k}\right)$ randomized estimation algorithm from block-multilinear polynomials to *arbitrary* bounded polynomials of degree k. As we said, Appendix 8 achieves this in the special case $k = 2$. Achieving it for arbitrary k seems likely to require generalizing the machinery of Dinur et al. [14].

A third problem concerns the notion of block-multilinear approximate degree, $\widetilde{\operatorname{bmdeg}}(f)$, that we introduced to prove Theorem 4. Is there any asymptotic separation between $\widetilde{\operatorname{bmdeg}}(f)$ and ordinary approximate degree? What about a separation between $\widetilde{\operatorname{bmdeg}}(f)$ and quantum query complexity?

A fourth, more open-ended problem is whether there are any applications of Forrelation, in the same sense that factoring and discrete log provide "applications" of Shor's period-finding problem. Concretely, are there any situations where one has two efficiently-computable Boolean functions $f, g : \{0,1\}^n \to \{-1,1\}$ (described, for example, by circuits), one wants to estimate how forrelated they are, *and* the structure of f and g does not provide a fast classical way to do this?

Here are five other open problems:

(1) Can we tighten the lower bound on the randomized query complexity of Forrelation from $\Omega(\frac{\sqrt{N}}{\log N})$ to $\Omega(\sqrt{N})$, or give an $O(\frac{\sqrt{N}}{\log N})$ upper bound?

(2) Can we generalize our results from Boolean to non-Boolean functions?

(3) What are the largest possible quantum versus classical query complexity separations for *sampling* problems? Is an $O(1)$ versus $\Omega(N)$ separation possible in this case? Also, what separations are possible for search or relation problems? (For our results on these questions, see Appendix 7.)

(4) While there exists a 1-query quantum algorithm that solves Forrelation with bounded error probability, the error probability we are able to achieve is about 0.4—more than the customary 1/3. If we want (say) a 1 versus $N^{\Omega(1)}$ quantum versus classical query complexity separation, then how small can the quantum algorithm's error be?

(5) While we show in Appendix 10 that being "*un*forrelated"—that is, having $\Phi_{f,g} \leq \frac{1}{100}$—behaves nicely as a property-testing problem, it would be interesting to show the same for being forrelated.

2 Acknowledgments

We are grateful to Ronald de Wolf for early discussions, Andy Drucker for discussions about the randomized query complexity of $\operatorname{OR}(f, \ldots, f)$, Shalev Ben-David and Sean Hallgren for discussions about the glued-trees problem, and Saeed Mehraban for a numerical calculation.

References

[1] S. Aaronson. BQP and the polynomial hierarchy. In *Proc. ACM STOC*, 2010. arXiv:0910.4698.

[2] S. Aaronson. The equivalence of sampling and searching. In *Proc. Computer Science Symposium in Russia (CSR)*, 2011. arXiv:1009.5104, ECCC TR10-128.

[3] S. Aaronson and D. Gottesman. Improved simulation of stabilizer circuits. *Phys. Rev. A*, 70(052328), 2004. quant-ph/0406196.

[4] A. Ambainis. Polynomial degree vs. quantum query complexity. *J. Comput. Sys. Sci.*, 72(2):220–238, 2006. Earlier version in IEEE FOCS 2003. quant-ph/0305028.

[5] R. Beals, H. Buhrman, R. Cleve, M. Mosca, and R. de Wolf. Quantum lower bounds by polynomials. *J. ACM*, 48(4):778–797, 2001. Earlier version in IEEE FOCS 1998, pp. 352-361. quant-ph/9802049.

[6] J. N. de Beaudrap, R. Cleve, and J. Watrous. Sharp quantum versus classical query complexity separations. *Algorithmica*, 34(4):449–461, 2002. quant-ph/0011065.

[7] C. Bennett, E. Bernstein, G. Brassard, and U. Vazirani. Strengths and weaknesses of quantum computing. *SIAM J. Comput.*, 26(5):1510–1523, 1997. quant-ph/9701001.

[8] H. Buhrman, L. Fortnow, I. Newman, and H. Röhrig. Quantum property testing. *SIAM J. Comput.*, 37(5):1387–1400, 2008. Previous version in SODA'2003. quant-ph/0201117.

[9] H. Buhrman and R. de Wolf. Complexity measures and decision tree complexity: a survey. *Theoretical Comput. Sci.*, 288:21–43, 2002.

[10] S. Chakraborty, E. Fischer, A. Matsliah, and R. de Wolf. New results on quantum property testing. In *Proc. Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, pages 145–156, 2010. arXiv:1005.0523.

[11] A. M. Childs, R. Cleve, E. Deotto, E. Farhi, S. Gutmann, and D. A. Spielman. Exponential algorithmic speedup by a quantum walk. In *Proc. ACM STOC*, pages 59–68, 2003. quant-ph/0209131.

[12] R. Cleve. The query complexity of order-finding. *Inf. Comput.*, 192(2):162–171, 2004. Earlier version in CCC'2000. quant-ph/9911124.

[13] R. Cleve and J. Watrous. Fast parallel circuits for the quantum Fourier transform. In *Proc. IEEE FOCS*, pages 526–536, 2000. quant-ph/0006004.

[14] I. Dinur, E. Friedgut, G. Kindler, and R. O'Donnell. On the Fourier tails of bounded functions over the discrete cube. In *Proc. ACM STOC*, pages 437–446, 2006.

[15] A. Drucker. Improved direct product theorems for randomized query complexity. *Computational Complexity*, 21(2):197–244, 2012. Earlier version in CCC'2011. ECCC TR10-080.

[16] S. Fenner and Y. Zhang. A note on the classical lower bound for a quantum walk algorithm. quant-ph/0312230v1, 2003.

[17] A. Montanaro and R. de Wolf. A survey of quantum property testing. arXiv:1310.2035, 2013.

[18] M. Nielsen and I. Chuang. *Quantum Computation and Quantum Information.* Cambridge University Press, 2000.

[19] B. Reichardt. Reflections for quantum query algorithms. In *Proc. ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 560–569, 2011. arXiv:1005.1601.

[20] O. Shamir. A variant of Azuma's inequality for martingales with subgaussian tails. arXiv:1110.2392, 2011.

[21] Y. Shi. Both Toffoli and controlled-NOT need little help to do universal quantum computation. *Quantum Information and Computation*, 3(1):84–92, 2002. quant-ph/0205115.

[22] P. W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, 26(5):1484–1509, 1997. Earlier version in IEEE FOCS 1994. quant-ph/9508027.

[23] D. Simon. On the power of quantum computation. In *Proc. IEEE FOCS*, pages 116–123, 1994.

3 Preliminaries

We assume familiarity with basic concepts of quantum computing, as covered (for example) in Nielsen and Chuang [18]. We also assume some familiarity with the model of query or decision-tree complexity; see Buhrman and de Wolf [9] for a good survey. In this section, we first give a brief recap of query complexity (in Section 3.1), then observe some properties of the k-fold Forrelation problem (in Section 3.2), and finally collect some lemmas about Gram-Schmidt orthogonalization (in Section 3.3) and Gaussian martingales (in Section 3.4) that will be important for our randomized lower bound in Section 4.

3.1 Query Complexity

Briefly, by the *query complexity* of an algorithm $\mathcal{A}$, we mean the number of queries that $\mathcal{A}$ makes to its input $z = (z_1, \ldots, z_N)$, maximized over all valid inputs z.[9] The query complexity of a function F is then the minimum query complexity of *any* algorithm $\mathcal{A}$ (of a specified type—classical, quantum, etc.) that outputs $F(z)$, with bounded probability of error, given any valid input z.

One slightly unconventional choice that we make is to define "bounded probability of error" to mean "error probability at most $1/2 - \varepsilon$, for some constant $\varepsilon > 0$" rather than "error probability at most $1/3$." The reason is that we will be able to design a 1-query quantum algorithm that solves the Forrelation problem with error probability $2/5$, but *not* one that solves it with error probability $1/3$. Of course, one can make the error probability as small as one likes using amplification, but doing so increases the query complexity by a constant factor.

We assume throughout this paper that the input $z \in \{-1, 1\}^N$ is Boolean, and we typically work in the $\{-1, 1\}$ basis for convenience. In the classical setting, each query returns a single bit z_i, for some index $i \in [N]$ specified by $\mathcal{A}$. In the quantum setting, each query performs a diagonal unitary transformation

$$|i, w\rangle \rightarrow z_i |i, w\rangle,$$

where w represents "workspace qubits" that do not participate in the query.[10] Between two queries, $\mathcal{A}$ can apply any unitary transformation it likes that does not depend on z.

[9] If we are talking about a partial Boolean function, then a "valid" input is simply any input that satisfies the promise.

[10] For Boolean inputs z, this is well-known to be exactly equivalent to a different definition of a quantum query, wherein each basis state $|i, a, w\rangle$ gets mapped to $|i, a \oplus z_i, w\rangle$. Here a represents a 1-qubit "answer register."

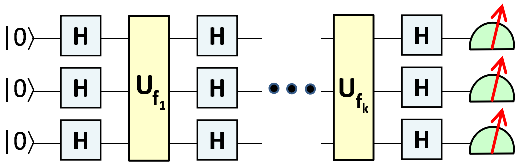

Figure 1: A quantum circuit that can be taken to define the k-fold Forrelation problem. The circuit consists of k query transformations $U_{f_1}, \ldots, U_{f_k}$, which map each basis state $|x\rangle$ to $f_i(x)|x\rangle$, sandwiched between rounds of Hadamard gates.

In this paper, the input $z = (z_1, \ldots, z_N)$ will typically consist of the truth tables of one or more Boolean functions: for example, $f, g : \{0,1\}^n \to \{-1,1\}$, or $f_1, \ldots, f_k : \{0,1\}^n \to \{-1,1\}$. Throughout, we use n for the number of input bits that these Boolean functions take (which roughly corresponds to the number of *qubits* in a quantum algorithm), and we use $N = 2^n$ for the number of bits being queried in superposition. (Strictly speaking, we should set $N = k \cdot 2^n$, where k is the number of Boolean functions. But this constant-factor difference will not matter for us.) Thus, for the purposes of query complexity, N is the "input size," in terms of which we state our upper and lower bounds.

3.2 Forrelation

The Forrelation and k-fold Forrelation problems were defined in Sections 1.1.1 and 1.1.3 respectively. Informally, though, one could define k-fold Forrelation simply as the problem of simulating the quantum circuit shown in Figure 1—and in particular, of estimating the amplitude, call it $\alpha_{0\cdots0}$, with which this circuit returns $|0\rangle^{\otimes n}$ as its output. Observe that $\alpha_{0\cdots0}$ is *precisely* the quantity

$$\Phi_{f_1,\ldots,f_k} := \frac{1}{2^{(k+1)n/2}} \sum_{x_1,\ldots,x_k \in \{0,1\}^n} f_1(x_1)(-1)^{x_1 \cdot x_2} f_2(x_2)(-1)^{x_2 \cdot x_3} \cdots (-1)^{x_{k-1} \cdot x_k} f_k(x_k)$$

defined in Section 1.1.3. From this, it follows that we can decide whether $|\Phi_{f_1,\ldots,f_k}| \leq \frac{1}{100}$ or $\Phi_{f_1,\ldots,f_k} \geq \frac{3}{5}$ with bounded probability of error, and thereby solve the k-fold Forrelation problem, by making only k quantum queries to $f_1, \ldots, f_k$.

Slightly more interesting is that we can improve the quantum query complexity further, to $\lceil k/2 \rceil$:

Proposition 6 *The k-fold* Forrelation *problem is solvable, with error probability* 0.4*, using $\lceil k/2 \rceil$ quantum queries to the functions $f_1, \ldots, f_k : \{0,1\}^n \to \{-1,1\}$, as well as $O(nk)$ quantum gates.*

Proof. Let H be the Hadamard gate, and let U_{f_i} be the query transformation that maps each computational basis state $|x\rangle$ to $f_i(x)|x\rangle$. Then to improve from k to $\lceil k/2 \rceil$ queries, we modify the circuit of Figure 1 in the following way.

In addition to the initial state $|0\rangle^{\otimes n}$, we prepare a control qubit in the state $|+\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}}$. Then, conditioned on the control qubit being $|0\rangle$, we apply the following sequence of operations to the initial state:

$$\mathrm{H}^{\otimes n} \to U_{f_1} \to \mathrm{H}^{\otimes n} \to U_{f_2} \to \cdots \to \mathrm{H}^{\otimes n} \to U_{f_{\lceil k/2 \rceil}} \to \mathrm{H}^{\otimes n}.$$

Meanwhile, conditioned on the control qubit being $|1\rangle$, we apply the following sequence of operations:

$$\mathrm{H}^{\otimes n} \to U_{f_k} \to \mathrm{H}^{\otimes n} \to U_{f_{k-1}} \to \cdots \to \mathrm{H}^{\otimes n} \to U_{f_{\lceil k/2 \rceil+1}}.$$

Finally, we measure the control qubit in the $\{|+\rangle, |-\rangle\}$ basis, and "accept" (i.e., say that $\Phi_{f_1,\ldots,f_k}$ is large) if and only if we find it in the state $|+\rangle$.

It is not hard to see that the probability that this circuit accepts is exactly

$$\frac{1+\Phi_{f_1,\ldots,f_k}}{2}.$$

Thus, consider an algorithm $\mathcal{A}$ that rejects with probability 1/4, and runs the circuit with probability 3/4. We have

$$\Pr[\mathcal{A} \text{ accepts}] = \frac{3}{4}\left(\frac{1+\Phi_{f_1,\ldots,f_k}}{2}\right).$$

If $|\Phi_{f_1,\ldots,f_k}| \leq \frac{1}{100}$ then the above is less than 0.4, while if $\Phi_{f_1,\ldots,f_k} \geq \frac{3}{5}$ then it is at least 0.6. ■

Purely from the unitarity of the quantum algorithm to compute $\Phi_{f_1,\ldots,f_k}$, we can derive some interesting facts about $\Phi_{f_1,\ldots,f_k}$ itself. Most obviously, we have $|\Phi_{f_1,\ldots,f_k}| \leq 1$. But beyond that, let $f_k^{(x)}(x_k) := f_k(x_k)(-1)^{x_k \cdot x}$. Then

$$\sum_{x \in \{0,1\}^n} \Phi^2_{f_1,\ldots,f_{k-1},f_k^{(x)}} = 1; \tag{1}$$

this is just saying that the sum of the squares of the final amplitudes in the Forrelation algorithm must be 1. Since there is nothing "special" about the outcome $|0\cdots 0\rangle$, it follows by symmetry that

$$\mathrm{E}\left[\Phi^2_{f_1,\ldots,f_k}\right] = \frac{1}{N}$$

if $f_1, \ldots, f_k$ are chosen are uniformly at random.

3.3 Gram-Schmidt Orthogonalization

Given an arbitrary collection of linearly-independent unit vectors $|v_1\rangle, |v_2\rangle \ldots$, the *Gram-Schmidt process* produces orthonormal vectors by recursively projecting each $|v_i\rangle$ onto the orthogonal complement of the subspace spanned by $|v_1\rangle, \ldots, |v_{i-1}\rangle$, and then normalizing the result. That is:

$$|z_i\rangle = |v_i\rangle - \sum_{j=1}^{i-1} \langle v_i|w_j\rangle |w_j\rangle,$$
$$|w_i\rangle = \beta_i |z_i\rangle$$

where $\beta_i = \frac{1}{\sqrt{\langle z_i|z_i\rangle}}$ is a normalizing constant. Note that $\langle z_i|z_i\rangle \leq 1$ (since $|z_i\rangle$ is the projection of a unit vector onto a subspace), and hence $\beta_i \geq 1$.

We will be interested in the behavior of this process when the $|v_i\rangle$'s are already very close to orthogonal. We can control that behavior with the help of the following lemma.

Lemma 7 (Gram-Schmidt Lemma) *Let $|v_1\rangle, \ldots, |v_t\rangle$ be unit vectors with $|\langle v_i|v_j\rangle| \leq \varepsilon$ for all $i \neq j$, and suppose $t \leq 0.1/\varepsilon$ (so in particular, $\varepsilon \leq 0.1$). Let $|w_i\rangle$ and β_i be as above. Then for all $i > j$, we have*

$$\begin{aligned} |\langle v_i|w_j\rangle| &\leq \varepsilon + 2j\varepsilon^2, \\ \beta_i &\leq 1 + 2i\varepsilon^2. \end{aligned}$$

So in particular, under the stated hypothesis, $|\langle v_i|w_j\rangle| \leq 1.2\varepsilon$ and $\beta_i \leq 1 + 0.2\varepsilon$.

Proof. We will do an induction on ordered pairs (i, j), in the order $(2,1), (3,1), (3,2), (4,1), \ldots$, with two induction hypotheses. Here are the hypotheses: for all $i > j$,

$$\begin{aligned} |\langle v_i|w_j\rangle| &\leq \varepsilon + Aj\varepsilon^2, \\ 1 - \langle z_i|z_i\rangle &\leq Bi\varepsilon^2. \end{aligned}$$

for some constants A, B to be determined later.

For the base case ($i = 2$ and $j = 1$), we have $|\langle v_2|w_1\rangle| = |\langle v_2|v_1\rangle| \leq \varepsilon$ and

$$\begin{aligned} \langle z_2|z_2\rangle &= (\langle v_2| - \langle v_1|v_2\rangle \langle v_1|)(|v_2\rangle - \langle v_1|v_2\rangle |v_1\rangle) \\ &= 1 - \langle v_1|v_2\rangle^2 \\ &\geq 1 - \varepsilon^2. \end{aligned}$$

For the induction step: first,

$$\begin{aligned} \langle v_i|w_j\rangle &= \langle v_i| \beta_j \left(|v_j\rangle - \sum_{k=1}^{j-1} \langle v_j|w_k\rangle |w_k\rangle \right) \\ &= \frac{1}{\sqrt{\langle z_j|z_j\rangle}} \left(\langle v_i|v_j\rangle + \sum_{k=1}^{j-1} \langle v_i|w_k\rangle \langle v_j|w_k\rangle \right). \end{aligned}$$

So

$$\begin{aligned} |\langle v_i|w_j\rangle| &\leq \frac{1}{\sqrt{1 - Bj\varepsilon^2}} \left(\varepsilon + \sum_{k=1}^{j-1} \left(\varepsilon + Ak\varepsilon^2\right)^2 \right) \\ &\leq \frac{1}{1 - Bj\varepsilon^2} \left(\varepsilon + j\varepsilon^2 + Aj^2\varepsilon^3 + \frac{A^2 j^3 \varepsilon^4}{3} \right) \\ &\leq \left(1 + \frac{B}{1 - 0.01B} j\varepsilon^2 \right) \left(\varepsilon + \left(1 + 0.1A + \frac{0.01A^2}{3} \right) j\varepsilon^2 \right) \\ &\leq \varepsilon + \left[\left(1 + 0.1A + \frac{0.01A^2}{3} \right) \left(1 + \frac{0.01B}{1 - 0.01B} \right) + \frac{0.1B}{1 - 0.01B} \right] j\varepsilon^2, \end{aligned}$$

where we repeatedly made the substitutions $\varepsilon \leq 0.1$ and $j\varepsilon \leq 0.1$ to produce multiples of $j\varepsilon^2$ in the numerator, and get rid of j and ε in the denominator. Second,

$$\begin{aligned}\langle z_i|z_i\rangle &= \left(\langle v_i| - \sum_{j=1}^{i-1} \langle v_i|w_j\rangle \langle w_j|\right)\left(|v_i\rangle - \sum_{j=1}^{i-1} \langle v_i|w_j\rangle |w_j\rangle\right) \\ &= \langle v_i|v_i\rangle - \sum_{j=1}^{i-1} \langle v_i|w_j\rangle^2 + \sum_{j\neq k\in[i-1]} \langle v_i|w_j\rangle \langle v_i|w_k\rangle \langle w_j|w_k\rangle \\ &= 1 - \sum_{j=1}^{i-1} \langle v_i|w_j\rangle^2\end{aligned}$$

where we used the fact that $\langle w_j|w_k\rangle = 0$. So

$$\begin{aligned}1 - \langle z_i|z_i\rangle &\leq \sum_{j=1}^{i-1} \left(\varepsilon + Aj\varepsilon^2\right)^2 \\ &\leq i\varepsilon^2 + Ai^2\varepsilon^3 + \frac{A^2 i^3 \varepsilon^4}{3} \\ &\leq \left(1 + 0.1A + \frac{0.01A^2}{3}\right) i\varepsilon^2.\end{aligned}$$

If we now make the choice (say) $A = 2$ and $B = 1.5$, we find that both parts of the induction are satisfied:

$$\begin{aligned}|\langle v_i|w_j\rangle| &\leq \varepsilon + 1.39j\varepsilon^2 \leq \varepsilon + Aj\varepsilon^2, \\ 1 - \langle z_i|z_i\rangle &\leq 1.22i\varepsilon^2 \leq Bi\varepsilon^2.\end{aligned}$$

Furthermore, we now have the lemma, since

$$\beta_i = \frac{1}{\sqrt{\langle z_i|z_i\rangle}} \leq \frac{1}{\langle z_i|z_i\rangle} = 1 + \frac{1 - \langle z_i|z_i\rangle}{\langle z_i|z_i\rangle} \leq 1 + \frac{Bi\varepsilon^2}{1 - Bi\varepsilon^2} \leq 1 + \frac{Bi\varepsilon^2}{1 - 0.01B} \leq 1 + 2i\varepsilon^2.$$

■

3.4 Gaussian Azuma's Inequality

Azuma's inequality is a well-known generalization of the Chernoff/Hoeffing tail bound to the case of martingales with bounded differences. We will need a generalization of Azuma's inequality to martingale difference sequences in which each term is Gaussian (and therefore, unbounded). Fortunately, Shamir [20, Theorem 2] recently proved a useful such generalization. We now state Shamir's bound, in a slightly different form than in [20] (but easily seen to be equivalent).

Lemma 8 (Gaussian Azuma's Inequality [20]) *Suppose $x_1, x_2, \ldots$ form a martingale difference sequence, in the sense that* $\mathrm{E}[x_i|x_1, \ldots, x_{i-1}] = 0$. *Suppose further that, conditioned on its predecessors, x_i is always "dominated by an $\mathcal{N}(0, \sigma^2)$ Gaussian," in the sense that* $\Pr[|x_i| > \sigma B] < \exp(-B^2/2)$ *for all B. Then*

$$\Pr\left[|x_1 + \cdots + x_t| > c\sigma\sqrt{t}\right] < 2\exp\left(-\frac{c^2}{56}\right).$$

Note, in particular, that if the x_i's themselves are $\mathcal{N}\left(0,\tau_i^2\right)$ Gaussians for some (possibly-differing) variances $\tau_i < \sigma$, then the x_i's are dominated by $\mathcal{N}\left(0,\sigma^2\right)$ Gaussians, so Lemma 8 can be applied.

4 Maximal Quantum/Classical Query Complexity Gap

In this section, we prove that the randomized query complexity of FORRELATION is $\Omega(\frac{\sqrt{N}}{\log N})$. Previously, Aaronson [1] proved an $\Omega(N^{1/4})$ randomized lower bound for this problem. We will need a further idea to improve the lower bound to $\widetilde{\Omega}(N^{1/3})$, a still further idea to improve it to $\widetilde{\Omega}(N^{2/5})$, and then yet another idea to get all the way up to $\widetilde{\Omega}(\sqrt{N})$.

Following [1], the first step is to replace FORRELATION by a "continuous relaxation" of the problem: a version that is strictly easier (and thus, for which proving a lower bound is *harder*), but which has rotational symmetry that will be extremely convenient for us. Thus, in REAL FORRELATION, we are given oracle access to two real functions $f,g:\{0,1\}^n \to \mathbb{R}$. As usual, the "input size" is $N=2^n$. We are promised that the pair $\langle f,g\rangle$ was drawn from one of two probability measures:

(i) In the uniform measure $\mathcal{U}$, every $f(x)$ and $g(y)$ value is an independent $\mathcal{N}(0,1)$ Gaussian.

(ii) In the forrelated measure $\mathcal{F}$, every $f(x)$ value is an independent $\mathcal{N}(0,1)$ Gaussian, while every $g(y)$ value is fixed to

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

The problem is to decide, with constant bias, whether (i) or (ii) holds (i.e., whether $\langle f,g\rangle$ was drawn from $\mathcal{U}$ or from $\mathcal{F}$).

We will often treat (the truth tables of) f and g as vectors in $\mathbb{R}^N$. Then another way to think about REAL FORRELATION is this: in case (i), f and g are drawn independently from $\mathcal{N}(0,1)^N$. In case (ii), f and g are *also* both distributed according to $\mathcal{N}(0,1)^N$, by the rotational symmetry of the Gaussian measure and the unitarity of the Hadamard transform. But they are no longer independent: they are related by $g=Hf$, where H is the $N\times N$ Hadamard matrix, given by $H_{x,y}=(-1)^{x\cdot y}/\sqrt{N}$. The problem is to detect whether this correlation is present.

An algorithm for REAL FORRELATION proceeds by querying $f(x)$ and $g(y)$ values one at a time, deciding which x or y to query next based on the values seen so far. We are interested in the expected number of queries needed by the best algorithm.

4.1 Discrete Versus Continuous

As a first step, we need to show that a lower bound for REAL FORRELATION really does imply the same lower bound for the original, Boolean FORRELATION problem. The key to doing so is the following result, which calculates the expected value of $\Phi_{F,G}$, for Boolean FORRELATION instances $\langle F,G\rangle$ that are produced by "rounding" real instances $\langle f,g\rangle$ in a natural way.

Theorem 9 *Suppose $\langle f, g\rangle$ are drawn from the forrelated measure $\mathcal{F}$. Define Boolean functions $F, G : \{0,1\}^n \to \{-1,1\}$ by $F(x) := \operatorname{sgn}(f(x))$ and $G(y) := \operatorname{sgn}(g(y))$. Then*

$$\mathrm{E}_{f,g\sim\mathcal{F}}\left[\Phi_{F,G}\right] = \frac{2}{\pi} \pm O\left(\frac{\log N}{N}\right).$$

Proof. By linearity of expectation, it suffices to calculate $\mathrm{E}\left[F(x)(-1)^{x\cdot y} G(y)\right]$ for some *specific* x, y pair. Let $v \in \mathbb{R}^N$ be a vector of independent $\mathcal{N}(0,1)$ Gaussians, and let H be the $N \times N$ Hadamard matrix (without normalization), with entries $H_{x,y} = (-1)^{x\cdot y}$. Then we can consider $\langle F, G\rangle$ to have been generated as follows:

$$F(x) = \operatorname{sgn}(v_x),$$
$$G(y) = \operatorname{sgn}((Hv)_y).$$

Now, $(Hv)_y$ can be expressed as the sum of $H_{x,y}v_x$ with the independent Gaussian random variable

$$Z := \sum_{x'\neq x} H_{x',y} v_{x'}.$$

Let $G'(y) := \operatorname{sgn}(Z)$. Then

$$\mathrm{E}\left[F(x)(-1)^{x\cdot y} G'(y)\right] = \mathrm{E}\left[\operatorname{sgn}(v_x)(-1)^{x\cdot y}\operatorname{sgn}(Z)\right] = 0,$$

since v_x and Z are independent Gaussians both with mean 0. Note that adding $H_{x,y}v_x$ back to Z can only flip Z to having the *same* sign as $\operatorname{sgn}(v_x)(-1)^{x\cdot y}$, not the opposite sign—and hence can only increase $F(x)(-1)^{x\cdot y} G(y)$. It follows that

$$\mathrm{E}\left[F(x)(-1)^{x\cdot y} G(y)\right] = 2\Pr\left[G(y) \neq G'(y)\right].$$

The event $G(y) \neq G'(y)$ occurs if and only if the following two events both occur:

$$|H_{x,y}v_x| > |Z|,$$
$$\operatorname{sgn}(H_{x,y}v_x) \neq \operatorname{sgn}(Z).$$

Since $H_{x,y} \in \{-1,1\}$ and the distribution of v_x is symmetric about 0, we can assume without loss of generality that $H_{x,y} = 1$.

Let $Z(t)$ be the probability density function of Z. Then

$$\begin{aligned}\Pr\left[|H_{x,y}v_x| > |Z| \quad\text{and}\quad \operatorname{sgn}(H_{x,y}v_x) \neq \operatorname{sgn}(Z)\right] &= 2\int_{t=0}^{\infty} Z(t)\Pr\left[H_{x,y}v_x > t\right] dt \\ &= 2\int_{t=0}^{\infty} Z(t)\Pr\left[v_x > t\right] dt.\end{aligned}$$

(Here the factor of 2 appears because we are restricting to the case $Z > 0$, and there is an equal probability coming from the $Z < 0$ case.)

As a linear combination of $N-1$ independent $\mathcal{N}(0,1)$ Gaussians, with ± 1 coefficients, Z has the $\mathcal{N}(0,N-1)$ Gaussian distribution. Therefore

$$\begin{aligned}
2\int_{t=0}^{\infty} Z(t)\Pr[v_x>t]\,dt &= \frac{2}{\sqrt{2\pi(N-1)}}\int_{t=0}^{\infty}\exp\left(-\frac{t^2}{2(N-1)}\right)\Pr[v_x>t]\,dt \\
&\leq \frac{2}{\sqrt{2\pi(N-1)}}\int_{t=0}^{\infty}\Pr[v_x>t]\,dt \\
&= \frac{2}{\sqrt{2\pi(N-1)}}\operatorname{E}[|v_x|] \\
&= \frac{2}{\pi\sqrt{N-1}} \\
&\leq \frac{2}{\pi\sqrt{N}}+O\left(\frac{1}{N^{3/2}}\right).
\end{aligned}$$

Here the fourth line follows from $\operatorname{E}[|X|]=\sqrt{2/\pi}$ when X is an $\mathcal{N}(0,1)$ Gaussian. In the other direction, for all $C>0$ we have

$$\begin{aligned}
2\int_{t=0}^{\infty} Z(t)\Pr[v_x>t]\,dt &= \frac{2}{\sqrt{2\pi(N-1)}}\int_{t=0}^{\infty}\exp\left(-\frac{t^2}{2(N-1)}\right)\Pr[v_x>t]\,dt \\
&\geq \frac{2}{\sqrt{2\pi N}}\int_{t=0}^{C}\exp\left(-\frac{t^2}{2(N-1)}\right)\Pr[v_x>t]\,dt \\
&\geq \frac{2}{\sqrt{2\pi N}}\exp\left(-\frac{C^2}{2(N-1)}\right)\int_{t=0}^{C}\Pr[v_x>t]\,dt \\
&= \frac{2}{\sqrt{2\pi N}}\exp\left(-\frac{C^2}{2(N-1)}\right)\left(\operatorname{E}[|v_x|]-\frac{1}{\sqrt{2\pi}}\int_{t=C}^{\infty}te^{-t^2/2}dt\right) \\
&= \frac{2}{\sqrt{2\pi N}}\exp\left(-\frac{C^2}{2(N-1)}\right)\left(\sqrt{\frac{2}{\pi}}-\frac{e^{-C^2/2}}{\sqrt{2\pi}}\right).
\end{aligned}$$

If we set $C:=\sqrt{\log N}$, then the above is

$$\frac{2}{\sqrt{2\pi N}}\left(1-O\left(\frac{\log N}{N}\right)\right)\left(\sqrt{\frac{2}{\pi}}-\frac{1}{\sqrt{2\pi N}}\right)\geq\frac{2}{\pi\sqrt{N}}-O\left(\frac{\log N}{N^{3/2}}\right).$$

Therefore

$$\begin{aligned}
\operatorname{E}[\Phi_{F,G}] &= \frac{1}{2^{3n/2}}\sum_{x,y\in\{0,1\}^n}\operatorname{E}\left[F(x)(-1)^{x\cdot y}G(y)\right] \\
&= \frac{N^2}{N^{3/2}}\cdot\left(\frac{2}{\pi\sqrt{N}}\pm O\left(\frac{\log N}{N^{3/2}}\right)\right) \\
&= \frac{2}{\pi}\pm O\left(\frac{\log N}{N}\right).
\end{aligned}$$

■

Earlier, Aaronson [1, Theorem 9] proved a variant of Theorem 9, but with a badly suboptimal constant: he was only able to show that

$$\mathrm{E}\left[\Phi_{F,G}\right] \geq \cos\left(2 \arccos \sqrt{\frac{2}{\pi}}\right) - o\left(1\right) \approx 0.273,$$

compared to the exact value of $2/\pi \approx 0.637$ that we get here. As a result, if we used [1], we would only be able to show hardness for distinguishing $\Phi_{F,G} \approx 0$ from (say) $\Phi_{F,G} \geq \frac{1}{4}$, rather than $\Phi_{F,G} \approx 0$ from $\Phi_{F,G} \geq \frac{3}{5}$.

We now use Theorem 9 to give the desired reduction from Real Forrelation to Forrelation.

Corollary 10 *Suppose there exists a T-query algorithm that solves* Forrelation *with bounded error. Then there also exists an $O\left(T\right)$-query algorithm that solves* Real Forrelation *with bounded error.*

Proof. Let $\langle f, g\rangle$ be an instance of Real Forrelation. Then we will produce an instance $\langle F, G\rangle$ of Boolean Forrelation exactly as in Theorem 9: that is, we set $F\left(x\right) := \operatorname{sgn}\left(f\left(x\right)\right)$ for all x and $G\left(y\right) := \operatorname{sgn}\left(g\left(y\right)\right)$ for all y. If $\langle f, g\rangle$ was drawn from the uniform measure $\mathcal{U}$, then $\mathrm{E}\left[\Phi_{F,G}^2\right] = \frac{1}{N}$ by symmetry. So by Markov's inequality,

$$\Pr\left[\left|\Phi_{F,G}\right| > \frac{1}{100}\right] < \frac{10000}{N}.$$

By contrast, if $\langle f, g\rangle$ was drawn from the forrelated measure $\mathcal{F}$, then

$$\mathrm{E}\left[\Phi_{F,G}\right] \geq \frac{2}{\pi} - o\left(1\right)$$

by Theorem 9. By Markov's inequality (and the fact that $\Phi_{F,G} \leq 1$), it follows that for all constants $\varepsilon \in (0, 1/2)$,

$$\Pr\left[\Phi_{F,G} \geq \frac{2}{\pi} - \varepsilon\right] > \varepsilon.$$

So in particular,

$$\Pr\left[\Phi_{F,G} \geq \frac{3}{5}\right] > \frac{1}{30}.$$

Using a constant amount of amplification, we can clearly produce an $O\left(T\right)$-query algorithm for Forrelation that errs with probability at most (say) $\frac{1}{100}$ on all $\langle F, G\rangle$. By the union bound, such an algorithm distinguishes the case that $\langle f, g\rangle$ was drawn from $\mathcal{U}$ from the case that $\langle f, g\rangle$ was drawn from $\mathcal{F}$ with bias at least

$$\frac{1}{30} - \frac{10000}{N} - 2\left(\frac{1}{100}\right) = \Omega\left(1\right).$$

■

Because of Corollary 10, we see that, to prove a lower bound for Forrelation, it suffices to prove the same lower bound for Real Forrelation. Furthermore, because the Real Forrelation problem is to distinguish two probability distributions, we can assume without loss of generality that any algorithm for the latter is deterministic.

4.2 Lower Bound for Real Forrelation

We now proceed to a lower bound on the query complexity of Real Forrelation. As a first step, let us recast our problem more abstractly. For convenience, we will use ket notation ($|v\rangle$, $|w\rangle$, etc.) for vectors in $\mathbb{R}^N$, even if the vectors do not represent quantum states and are not even normalized. Let $|1\rangle,\ldots,|N\rangle$ be an orthonormal basis for $\mathbb{R}^N$, and let $|\hat{i}\rangle = H|i\rangle$ be the Hadamard transform of $|i\rangle$ (so that $|\hat{1}\rangle,\ldots,|\hat{N}\rangle$ is also an orthonormal basis).

Then consider the following generalization of Real Forrelation, which we call Gaussian Distinguishing. We are given a finite set $\mathcal{V}$ of unit vectors in $\mathbb{R}^N$, called "test vectors." (In our case, $\mathcal{V}$ happens to equal $\left\{|1\rangle,\ldots,|N\rangle,|\hat{1}\rangle,\ldots,|\hat{N}\rangle\right\}$.) In each step, we are allowed to pick any test vector $|v\rangle \in \mathcal{V}$ that hasn't been picked in previous steps. We then "query" $|v\rangle$, getting back a real-valued response $a_v \in \mathbb{R}$. The problem is to distinguish the following two cases, with constant bias:

(i) Each a_v is drawn independently from $\mathcal{N}(0,1)$.

(ii) Each a_v equals $\langle\Psi|v\rangle$, where $|\Psi\rangle \in \mathbb{R}^N$ is a vector drawn from $\mathcal{N}(0,1)^N$ that is fixed throughout the algorithm.

We will actually prove a *general* lower bound for Gaussian Distinguishing, which works whenever $|\mathcal{V}|$ is not too large, and every pair of vectors in $\mathcal{V}$ is sufficiently close to orthogonal. Here is our general result:

Theorem 11 *Suppose $|\mathcal{V}| \leq M$, and $|\langle v|w\rangle| \leq \varepsilon$ for all distinct vectors $|v\rangle,|w\rangle \in \mathcal{V}$. Then any classical algorithm for* Gaussian Distinguishing *must make $\Omega\left(\frac{1/\varepsilon}{\log(M/\varepsilon)}\right)$ queries.*

In our case (Real Forrelation), we have $M = 2N$ and $\varepsilon = 1/\sqrt{N}$, so the lower bound we get is $\Omega\left(\frac{\sqrt{N}}{\log N}\right)$. As a remark, the example of Real Forrelation shows that Theorem 11 is tight in its dependence on $1/\varepsilon$. One can also construct an example to show that the theorem's dependence on M is in some sense needed (if possibly not tight). That is, one does *not* have a $\widetilde{\Omega}(1/\varepsilon)$ lower bound on query complexity for arbitrarily large M, but at best a $\Omega((1/\varepsilon)^{2/3})$ lower bound.[11] In the context of Real Forrelation, this means that, if the only thing we knew about $\mathcal{V}$ was that $|\langle v|w\rangle| \leq 1/\sqrt{N}$ for all distinct $|v\rangle,|w\rangle \in \mathcal{V}$ (so in particular, we had no upper bound on $\mathcal{V}$'s cardinality), then we could not hope to prove any lower bound better than $\Omega(N^{1/3})$.[12]

[11] Here is the example that shows this: let $|1\rangle,\ldots,|n\rangle$ be orthogonal unit vectors. Then for all 2^n strings $z = z_1\cdots z_n \in \{-1,1\}^n$, let $|w_z\rangle$ be a vector such that $\langle w_z|i\rangle = z_i/n^{3/2}$ for all $i \in [n]$, and also such that the projections of the $|w_z\rangle$'s onto the orthogonal complement of $|1\rangle,\ldots,|n\rangle$ are all orthogonal to one another. Let $\mathcal{V} = \{|1\rangle,\ldots,|n\rangle\} \cup \{|w_z\rangle\}_{z\in\{-1,1\}^n}$. Then the inner product between any two distinct vectors in $\mathcal{V}$ is upper-bounded by $\varepsilon = 1/n^{3/2}$ in absolute value (the inner product between any two $|w_z\rangle$'s is at most $n/(n^{3/2})^2 = 1/n^2$). On the other hand, here is an algorithm that solves Gaussian Distinguishing using only $O(n) \ll 1/\varepsilon$ queries: first query $|1\rangle,\ldots,|n\rangle$ to obtain $a_1,\ldots,a_n$. Let $|\varphi\rangle := a_1|1\rangle + \cdots + a_n|n\rangle$. Next, find n distinct vectors $|w_z\rangle$ that each have inner product $\Theta\left(n/n^{3/2}\right) = \Theta(1/\sqrt{n})$ with $|\varphi\rangle$ (such vectors can always be found, so long as $|a_i| = \Omega(1)$ for $\Omega(n)$ values of i), and query all of them, letting $b_1,\ldots,b_n$ be the results. In case (i), we have $\mathrm{E}[b_1+\cdots+b_n]$ and $\mathrm{Var}[b_1+\cdots+b_n] = n$. But in case (ii), we have $\mathrm{E}[b_1+\cdots+b_n] = \Theta(\sqrt{n})$ and $\mathrm{Var}[b_1+\cdots+b_n] = O(n)$, allowing the two cases to be distinguished with constant bias.

[12] In fact one *can* prove a $\widetilde{\Omega}(N^{1/3})$ lower bound even under this restriction—and more generally, in the statement of Theorem 11, one can replace the lower bound $\Omega\left(\frac{1/\varepsilon}{\log(M/\varepsilon)}\right)$ by $\Omega\left(\frac{(1/\varepsilon)^{2/3}}{(\log 1/\varepsilon)^{1/3}}\right)$, independent of M. We will briefly remark on how to do this at the relevant point in our proof.

For the remainder of the proof, we will fix $\varepsilon = 1/\sqrt{N}$ for concreteness; but will leave M unfixed. Note that N will *only* enter into the proof through its relation with ε; the fact that N is also the dimensionality of the vectors will be irrelevant for us.[13]

The first question we need to answer is this: suppose an algorithm has queried test vectors $|v_1\rangle, \ldots, |v_t\rangle \in \mathcal{V}$, and has gotten back responses $a_1, \ldots, a_t \in \mathbb{R}$. Let $D = \{(|v_i\rangle, a_i)\}_i$ represent the data that the algorithm has seen. Then conditioned on D, how likely are we to be in case (i) or case (ii)? How much probability measure do $\mathcal{U}$ and $\mathcal{F}$ respectively assign to D?

For case (i), the question is easy to answer: the probability measure that $\mathcal{U}$ assigns to D is just the Gaussian one,

$$\mu_{\mathcal{U}}(D) = \frac{e^{-\Delta_{\mathcal{U}}(D)/2}}{(2\pi)^{t/2}},$$

where

$$\Delta_{\mathcal{U}}(D) := a_1^2 + \cdots + a_t^2$$

is the squared 2-norm of the vector of responses seen so far. For case (ii), by contrast, we start with $|\Psi\rangle$ drawn from $\mathcal{N}(0,1)^N$; then each data point restricts $|\Psi\rangle$ to the affine subspace S_i defined by $\langle \Psi | v_i \rangle = a_i$. Let $S(D) = S_1 \cap \cdots \cap S_t$ be the intersection of all these affine subspaces. Then the probability measure that $\mathcal{F}$ assigns to D is simply the measure that $\mathcal{N}(0,1)^N$ assigns to $S(D)$, which in turn (by rotational symmetry) is just the minimum squared 2-norm of any point in $S(D)$, scaled by a dimension factor. That is,

$$\mu_{\mathcal{F}}(D) = \frac{e^{-\Delta_{\mathcal{F}}(D)/2}}{(2\pi)^{t/2}},$$

where

$$\Delta_{\mathcal{F}}(D) := \min_{|\Phi\rangle \in S(D)} \langle \Phi | \Phi \rangle .$$

Putting the two things together, we have

$$\frac{\mu_{\mathcal{F}}(D)}{\mu_{\mathcal{U}}(D)} = \exp\left(\frac{\Delta_{\mathcal{U}}(D) - \Delta_{\mathcal{F}}(D)}{2} \right).$$

Thus, let

$$\Delta(D) := \Delta_{\mathcal{U}}(D) - \Delta_{\mathcal{F}}(D).$$

Then if we can just show that $|\Delta(D)|$ remains $o(1)$ after t queries, we will have shown that the algorithm cannot have distinguished case (i) from case (ii) with constant bias after t queries. Thus, upper-bounding $|\Delta(D)|$ will be our focus for the rest of the proof.

[13] By slightly modifying the example from footnote 11—to make the projections of the $|w_z\rangle$'s onto the orthogonal complement of $|1\rangle, \ldots, |n\rangle$ not exactly orthogonal to each other, but merely approximately orthogonal—one can produce an instance of Gaussian Distinguishing whose classical query complexity is only $O((1/\varepsilon)^{2/3})$, and which *also* satisfies $N = O(n^4) = O((1/\varepsilon)^{8/3})$. This is an exponential improvement in the dimensionality N compared to footnote 11. It would be interesting to know whether enforcing, say, $N = O((1/\varepsilon)^2)$ rules out such examples.

4.3 Upper-Bounding $|\Delta(D)|$

As a first observation, we cannot hope to show that $|\Delta(D)|$ remains small with *certainty*. Indeed, even after just 2 queries, $|\Delta(D)|$ could be unboundedly large, if the responses a_1 and a_2 were far out in the tails of $\mathcal{N}(0,1)$. Thus, our only hope is to show that, after few enough queries, $|\Delta(D)|$ remains small with *high probability*. But do we mean high probability with respect to $\mathcal{U}$ or $\mathcal{F}$? Crucially, we claim that the answer doesn't matter. To see this, suppose (for example) that we have $|\Delta(D)| = o(1)$ with probability $1 - o(1)$ over data D drawn according to $\mathcal{U}$. Then with probability $1 - o(1)$ over $\mathcal{U}$, we have

$$\frac{\mu_{\mathcal{F}}(D)}{\mu_{\mathcal{U}}(D)} = \exp\left(\frac{\Delta(D)}{2}\right) = \exp(\pm o(1)) = 1 \pm o(1).$$

It follows that we also have $|\Delta(D)| = o(1)$ with probability $1 - o(1)$ over data D drawn according to $\mathcal{F}$. So for simplicity, we will assume the data to be drawn according to $\mathcal{U}$.

Let us look more closely at the difference $\Delta(D) = \Delta_{\mathcal{U}}(D) - \Delta_{\mathcal{F}}(D)$. The $\Delta_{\mathcal{U}}(D)$ component is easy to compute, since it is just $a_1^2 + \cdots + a_t^2$. For the $\Delta_{\mathcal{F}}(D)$ component, on the other hand, we need to solve the linear-algebra problem of finding the distance between the affine subspace $S(D)$ and the origin. We can do this using Gram-Schmidt orthogonalization (see Section 3.3). That is, for each $i \in [t]$, we define $|w_i\rangle$ recursively as the normalized projection of $|v_i\rangle$ onto the orthogonal complement of the subspace spanned by $|w_1\rangle, \ldots, |w_{i-1}\rangle$. We can express $|w_i\rangle$ recursively as

$$|w_i\rangle = \beta_i \left(|v_i\rangle - \sum_{j=1}^{i-1} \langle v_i | w_j \rangle \, |w_j\rangle \right),$$

where β_i is a normalizing constant. Let us also define

$$\begin{aligned} b_i &= \langle \Psi | w_i \rangle \\ &= \beta_i \left(\langle \Psi | v_i \rangle - \sum_{j=1}^{i-1} \langle v_i | w_j \rangle \langle \Psi | w_j \rangle \right) \\ &= \beta_i \left(a_i - \sum_{j=1}^{i-1} \langle v_i | w_j \rangle \, b_j \right). \end{aligned}$$

Then we have:

$$\begin{aligned} \Delta_{\mathcal{F}}(D) &= \min_{|\Phi\rangle \in S(D)} \langle \Phi | \Phi \rangle \\ &= \min_{|\Phi\rangle \in S(D)} \sum_{i=1}^{t} \langle \Phi | w_i \rangle^2 \\ &= \sum_{i=1}^{t} \langle \Psi | w_i \rangle^2 \\ &= b_1^2 + \cdots + b_t^2 \end{aligned}$$

where the third line used the orthogonality of the $|w_i\rangle$'s.

To simplify matters, let us define a variant of b_i where we omit all the normalization factors β_i:

$$c_i := a_i - \sum_{j=1}^{i-1} \langle v_i | w_j \rangle \, c_j. \tag{2}$$

Also, call the data D *well-behaved* if $|a_i| \leq \sqrt{2 \ln 100t}$ for all $i \in [t]$.

Proposition 12 *D is well-behaved with probability at least 0.99 over $\mathcal{U}$.*

Proof. Follows from the union bound, together with the fact that each a_i is an independent $\mathcal{N}(0,1)$ Gaussian, so

$$\Pr\left[|a_i| > \sqrt{2 \ln 100t} \right] < \frac{1}{100t}.$$

■

Then we have the following useful lemma.

Lemma 13 *Let $t \leq \sqrt{N}/10$, and suppose D is well-behaved. Then $|c_i - b_i| = O\left(i \frac{\sqrt{\log N}}{N}\right)$ for all $i \in [t]$.*

Proof. If $|a_i| \leq \sqrt{2 \ln 100t}$ for all $i \in [t]$, then certainly $|b_i| = O(\sqrt{\log t}) = O(\sqrt{\log N})$ for all $i \in [t]$ as well, since

$$\begin{aligned} |b_i| &\leq \beta_i \left(|a_i| + \sum_{j=1}^{i-1} |\langle v_i | w_j \rangle| \, |b_j| \right) \\ &\leq \left(1 + \frac{0.2}{\sqrt{N}}\right) \left(|a_i| + \frac{0.2}{\sqrt{N}} \sum_{j=1}^{i-1} |b_j| \right) \\ &= O\left(\max_{j \in [i]} |a_j| \right), \end{aligned}$$

where the second line used Lemma 7 and the third used $i \leq t \leq \sqrt{N}/10$, together with induction on j. Now,

$$\begin{aligned} c_i - b_i &= a_i - \sum_{j=1}^{i-1} \langle v_i | w_j \rangle \, c_j - \beta_i \left(a_i - \sum_{j=1}^{i-1} \langle v_i | w_j \rangle \, b_j \right) \\ &= (1 - \beta_i) \, a_i - \sum_{j=1}^{i-1} \langle v_i | w_j \rangle \, (c_j - \beta_i b_j). \end{aligned}$$

So

$$\begin{aligned}
|c_i - b_i| &\leq (\beta_i - 1)\,|a_i| + \sum_{j=1}^{i-1} |\langle v_i | w_j \rangle| \left(|c_j - b_j| + (\beta_i - 1)\,|b_j| \right) \\
&\leq \frac{2i\,|a_i|}{N} + \frac{0.2}{\sqrt{N}} \sum_{j=1}^{i-1} \left(\frac{2i\,|b_j|}{N} + |c_j - b_j| \right) \\
&= O\left(\frac{i\sqrt{\log N}}{N} + \frac{i^2\sqrt{\log N}}{N^{3/2}} \right) + \frac{1}{\sqrt{N}} \sum_{j=1}^{i-1} |c_j - b_j| \\
&= O\left(\frac{i\sqrt{\log N}}{N} \right) + \frac{1}{\sqrt{N}} \sum_{j=1}^{i-1} |c_j - b_j|
\end{aligned}$$

where the second line used Lemma 7 and the last used $i \leq t \leq \sqrt{N}/10$. So, letting ε_i be an upper bound on $|c_j - b_j|$ for all $j \leq i$, we have

$$\begin{aligned}
\varepsilon_i &= O\left(\frac{i\sqrt{\log N}}{N} \right) + \frac{i}{\sqrt{N}} \varepsilon_{i-1} \\
&= O\left(\frac{i\sqrt{\log N}}{N} \right) + \frac{i}{\sqrt{N}} \varepsilon_i.
\end{aligned}$$

Rearranging, we have

$$0.9\varepsilon_i = O\left(\frac{i\sqrt{\log N}}{N} \right)$$

and are done. ■

As a first consequence of Lemma 13, if D is well-behaved, then

$$|c_i| = O\left(\sqrt{\log t} + i\frac{\sqrt{\log N}}{N} \right) = O(\sqrt{\log t})$$

for all $i \in [t]$. As a more important consequence, let

$$\Delta'_{\mathcal{F}}(D) := c_1^2 + \cdots + c_t^2,$$

and let

$$\Delta'(D) := \Delta_{\mathcal{U}}(D) - \Delta'_{\mathcal{F}}(D).$$

Then we can restrict our attention to upper-bounding $|\Delta'(D)|$, rather than the more complicated $|\Delta(D)|$. For by Lemma 13, if D is well-behaved, then

$$\begin{aligned}
\left| \Delta_{\mathcal{F}}(D) - \Delta'_{\mathcal{F}}(D) \right| &= \left| \sum_{i=1}^{t} \left(b_i^2 - c_i^2 \right) \right| \\
&\leq \sum_{i=1}^{t} \left(|b_i| + |c_i| \right) |c_i - b_i| \\
&= \sum_{i=1}^{t} O\left(\sqrt{\log N} \cdot i\frac{\sqrt{\log N}}{N} \right) \\
&= O\left(t^2 \frac{\log N}{N} \right).
\end{aligned}$$

So if $|\Delta'(D)| = o(1)$ and $t = o\left(\sqrt{\frac{N}{\log N}}\right)$, then by the triangle inequality,

$$|\Delta(D)| \leq \left|\Delta'(D)\right| + \left|\Delta_{\mathcal{F}}(D) - \Delta'_{\mathcal{F}}(D)\right|$$

is $o(1)$ as well. Thus, from now on our goal is to upper-bound $|\Delta'(D)|$.

Let

$$r_i := a_i - c_i = \sum_{j=1}^{i-1} \langle v_i | w_j \rangle \, c_j. \tag{3}$$

Notice that, if we unravel the recursive definition of c_j, we find that r_i is a linear combination of $a_1, \ldots, a_{i-1}$, with no dependence on a_i. Though we will not need this for the proof, r_i has an interesting interpretation, as the *expected* value of a_i after $a_1, \ldots, a_{i-1}$ have been queried but before a_i has been queried, assuming the data were drawn from the forrelated distribution $\mathcal{F}$. Now,

$$\begin{aligned}\Delta'(D) &= \Delta_{\mathcal{U}}(D) - \Delta'_{\mathcal{F}}(D) \\ &= \sum_{i=1}^{t} \left(a_i^2 - c_i^2\right) \\ &= \sum_{i=1}^{t} r_i \left(2a_i - r_i\right). \end{aligned} \tag{4}$$

As we show in the next lemma, the above means that our problem can in turn be reduced to upper-bounding the r_i's.

Lemma 14 *Suppose $|r_i| \leq \frac{1}{1750\sqrt{t}}$ for all $i \in [t]$. Then*

$$\left|\sum_{i=1}^{t} r_i a_i\right| \leq 0.01$$

with probability at least 0.99 *over the data D.*

Proof. Notice that each $r_i a_i$ has an expectation of 0, even after conditioning on $a_1, \ldots, a_{i-1}$. This is because, according to the measure $\mathcal{U}$, each a_i is a "fresh" $\mathcal{N}(0,1)$ Gaussian, uncorrelated with $a_1, \ldots, a_{i-1}$, whereas r_i is a linear combination of $a_1, \ldots, a_{i-1}$ that does not depend on a_i. Thus, $r_1 a_1, \ldots, r_t a_t$ forms a martingale difference sequence, in which, conditioned on its predecessors, each $r_i a_i$ is an $\mathcal{N}\left(0, r_i^2\right)$ Gaussian, for some $|r_i| \leq \frac{1}{1750\sqrt{t}}$. Set $\epsilon := 1/1750$. Then by Lemma 8,

$$\begin{aligned}\Pr\left[\left|\sum_{i=1}^{t} r_i a_i\right| > 0.01\right] &\leq \Pr\left[\left|\sum_{i=1}^{t} r_i a_i\right| > \sqrt{56 \ln 200} \frac{\epsilon}{\sqrt{t}} \sqrt{t}\right] \\ &< 2 \exp\left(-\frac{(\sqrt{56 \ln 200})^2}{56}\right) \\ &= 0.01. \end{aligned}$$

■

Thus, suppose $|r_i| \leq \frac{1}{1750\sqrt{t}}$ for all $i \in [t]$. Then by Lemma 14 and equation (4), we have

$$\begin{aligned}\left|\Delta'(D)\right| &= \left|\sum_{i=1}^{t} r_i\left(2a_i - r_i\right)\right| \\ &\leq 2\left|\sum_{i=1}^{t} r_i a_i\right| + \sum_{i=1}^{t} r_i^2 \\ &\leq 0.02 + \frac{1}{1750^2}\end{aligned}$$

with probability at least 0.99 over D. This implies that the algorithm has not yet succeeded at distinguishing $\mathcal{F}$ from $\mathcal{U}$ with bias (say) 1/2. So in summary, if we can show that with high probability, $|r_i| = O(1/\sqrt{t})$ for all $i \in [t]$, then we have shown that the algorithm must make $\Omega(t)$ queries.

4.4 Upper-Bounding $|r_i|$

We now turn to the problem of upper-bounding $|r_i|$ (with high probability over D), for all $i \in [t]$. The better the upper bound on $|r_i|$ we can achieve, the better will be our lower bound on t. To illustrate, it is easy to prove the following crude bound:

Proposition 15 *If D is well-behaved and $t < \sqrt{N}/10$, then $|r_i| = O\left(i\sqrt{\frac{\log N}{N}}\right)$ for all $i \in [t]$.*

Proof. We noted before that if D is well-behaved then $|c_i| = O(\sqrt{\log t}) = O(\sqrt{\log N})$ for all i. So by Lemma 7,

$$\begin{aligned}|r_i| &\leq \sum_{j=1}^{i-1} \left|\langle v_i | w_j \rangle\right| |c_j| \\ &\leq i \cdot \frac{2}{\sqrt{N}} \cdot O(\sqrt{\log N}).\end{aligned}$$

■

Setting $t\sqrt{\frac{\log N}{N}} = 1/\sqrt{t}$ and solving, Proposition 15 yields a lower bound of $t = \Omega\left(\left(\frac{N}{\log N}\right)^{1/3}\right)$ queries.[14]

With some more work, one can prove a bound of $|r_i| = O\left(\sqrt{\frac{i \log Mt}{N}} + \frac{i^2}{N}\right)$, which yields a lower bound of $t = \Omega\left(N^{2/5}\right)$ queries whenever $M \leq \exp\left(O\left(N^{1/5}\right)\right)$. In this section, however, we will go for the bound $|r_i| = O\left(\sqrt{\frac{i \log Mt}{N}}\right)$, which yields a lower bound of $t = \Omega\left(\frac{\sqrt{N}}{\log MN}\right)$ queries. For Real Forrelation, of course, we have $M = 2N$, and therefore $t = \Omega\left(\frac{\sqrt{N}}{\log N}\right)$ as desired.

Our strategy will be to make repeated use of the following lemma.

[14] Furthermore, notice that Proposition 15 has no dependence on the number of test vectors M. This is why it implies a $\Omega\left(\frac{(1/\varepsilon)^{2/3}}{(\log 1/\varepsilon)^{1/3}}\right)$ lower bound for Gaussian Distinguishing, independent of M.

Lemma 16 (Central Martingale Lemma) *Suppose* $200 \leq t < \sqrt{N}/10$. *Then with probability at least* 0.99 *over data* D *drawn from* $\mathcal{U}$, *we have*

$$\left| \sum_{j=1}^{i-1} \langle v|w_j \rangle \, a_j \right| \leq 30\sqrt{\frac{i \ln Mt}{N}}$$

for all $i \in [t]$ *and for all test vectors* $|v\rangle \in \mathcal{V}$.

Proof. Fix any $|v\rangle \in \mathcal{V}$. By Lemma 7, we have $|\langle v|w_j\rangle| \leq 2/\sqrt{N}$ for all $|v\rangle$ and $|w_j\rangle$. Also, recall that each a_j is a "fresh" $\mathcal{N}(0,1)$ Gaussian, and that $\langle v|w_j\rangle$ does not depend on a_j. Thus, $\langle v|w_1\rangle \, a_1, \ldots, \langle v|w_{i-1}\rangle \, a_{i-1}$ forms a martingale difference sequence, in which, conditioned on its predecessors, each $\langle v|w_j\rangle \, a_j$ is an $\mathcal{N}(0, \langle v|w_j\rangle^2)$ Gaussian. So by Lemma 8,

$$\begin{aligned}
\Pr\left[\left| \sum_{j=1}^{i-1} \langle v|w_j \rangle \, a_j \right| > 30\sqrt{\frac{i \ln Mt}{N}} \right] &< \Pr\left[\left| \sum_{j=1}^{i-1} \langle v|w_j \rangle \, a_j \right| > \sqrt{56 \ln(200Mt)} \frac{2}{\sqrt{N}} \sqrt{i} \right] \\
&< 2 \exp\left(-\frac{(\sqrt{56 \ln(200Mt)})^2}{56} \right) \\
&= \frac{1}{100Mt}.
\end{aligned}$$

The result now follows by taking the union bound over all $|v\rangle \in \mathcal{V}$ and $i \in [t]$. ■

We can now prove the desired upper bound on $|r_i|$.

Lemma 17 *Suppose* $t < \sqrt{N}/10$. *Then with probability at least* 0.99 *over* D, *we have* $|r_i| = O\left(\sqrt{\frac{i \log Mt}{N}}\right)$ *for all* $i \in [t]$.

Proof. Taking the equation for r_i (equation (3)), and unraveling the recursive definition of c_j (equation (2)), we get

$$\begin{aligned}
r_i &= \sum_{j=1}^{i-1} \langle v_i|w_j \rangle \, c_j \\
&= \sum_{j=1}^{i-1} \langle v_i|w_j \rangle \left(a_j - \sum_{k=1}^{j-1} \langle v_j|w_k \rangle \, c_k \right) \\
&= \sum_{j=1}^{i-1} \langle v_i|w_j \rangle \left(a_j - \sum_{k=1}^{j-1} \langle v_j|w_k \rangle \left(a_k - \sum_{\ell=1}^{k-1} \langle v_k|w_\ell \rangle \, c_\ell \right) \right) \\
&\vdots
\end{aligned}$$

Thus,

$$\begin{aligned}
|r_i| &\leq \left|\sum_{j=1}^{i-1}\langle v_i|w_j\rangle a_j\right| + \sum_{j=1}^{i-1}|\langle v_i|w_j\rangle|\left|\sum_{k=1}^{j-1}\langle v_j|w_k\rangle a_k\right| + \sum_{j=1}^{i-1}|\langle v_i|w_j\rangle|\sum_{k=1}^{j-1}|\langle v_j|w_k\rangle|\left|\sum_{\ell=1}^{k-1}\langle v_k|w_\ell\rangle a_\ell\right| + \cdots \\
&\leq 30\sqrt{\frac{i\ln Mt}{N}} + \sum_{j=1}^{i-1}\frac{2}{\sqrt{N}}30\sqrt{\frac{j\ln Mt}{N}} + \sum_{j=1}^{i-1}\sum_{k=1}^{j-1}\left(\frac{2}{\sqrt{N}}\right)^2 30\sqrt{\frac{k\ln Mt}{N}} + \cdots \\
&\leq 30\sqrt{\frac{\ln Mt}{N}}\left[\sqrt{i} + \frac{2}{\sqrt{N}}i^{3/2} + \left(\frac{2}{\sqrt{N}}\right)^2 i^{5/2} + \cdots\right] \\
&= 30\sqrt{\frac{i\ln Mt}{N}}\left[1 + \frac{2i}{\sqrt{N}} + \left(\frac{2i}{\sqrt{N}}\right)^2 + \cdots\right] \\
&= O\left(\sqrt{\frac{i\log Mt}{N}}\right)
\end{aligned}$$

where the second line used Lemmas 7 and 16. ■

5 Simulation of t-Query Quantum Algorithms

Let $\mathcal{A}$ be a quantum algorithm that makes $t = O(1)$ queries to a Boolean input $x \in \{-1,1\}^N$, and then either accepts or rejects. In this section, we show how to estimate $\mathcal{A}$'s acceptance probability, on all inputs x, by a classical, nonadaptive randomized algorithm that makes only $O(N^{1-1/2t})$ queries to x.

So for example, we can simulate any 1-query quantum algorithm using $O(\sqrt{N})$ classical queries—thereby showing that the 1 versus $\Omega(\frac{\sqrt{N}}{\log N})$ separation of Section 4 is nearly tight. More generally, resolving an open problem of Buhrman et al. [8], we find that there is *no* partial Boolean function whose quantum query complexity is constant but whose randomized query complexity is linear.

We obtain our simulation of quantum algorithms as a consequence of a much more general result: namely, that *any bounded, degree-k polynomial* $p : \{-1,1\}^N \to \mathbb{R}$*, which satisfies a technical condition called "block-multilinearity," can be estimated by querying only* $O(N^{1-1/k})$ *of its variables.* This result makes no direct reference to quantum computing, and seems likely to have independent applications—for example, to the design of classical sublinear algorithms. We strongly conjecture that the block-multilinearity condition can be removed, which would further heighten the non-quantum interest of this result. In Appendix 8, we prove that conjecture in the special case $k = 2$.

More formally, let $p : \mathbb{R}^N \to \mathbb{R}$ be a real polynomial of degree k. Since we will only care about p's behavior on the Boolean hypercube $\{-1,1\}^N$, we can assume without loss of generality that p is multilinear (that is, that no variable is raised to a higher power than 1). We call p *bounded* if $p(x) \in [-1,1]$ for all $x \in \{-1,1\}^N$.[15] Now, we call p *block-multilinear* if its N variables $x_1,\ldots,x_N$ can be partitioned into k blocks, $R_1,\ldots,R_k$, so that every monomial of p contains

[15] In quantum query complexity, normally we would call a polynomial p "bounded" if $p(x) \in [0,1]$ for all $x \in \{-1,1\}^N$—in other words, if p represents a probability. As we will see, however, we need to consider polynomials that can represent arbitrary inner products between vectors of norm at most 1, and can therefore only assume $p(x) \in [-1,1]$.

exactly one variable from each block. Note that block-multilinearity implies, in particular, that p is homogeneous. Also, by introducing at most $O(N)$ dummy variables, we can assume without loss of generality that every block has the same size, $|R_1| = \cdots = |R_k| = n = N/k$.

We can now state the main result of this section.

Theorem 18 *Let $p : \{-1,1\}^N \to [-1,1]$ be any bounded block-multilinear polynomial of degree k. Then there exists a classical randomized algorithm that, on input $x \in \{-1,1\}^N$, nonadaptively queries $O((N/\varepsilon^2)^{1-1/k})$ bits of x, and then outputs an estimate $\tilde{p}$ such that with high probability,*

$$|\tilde{p} - p(x_1, \ldots, x_N)| \leq \varepsilon.$$

(Here the big-O hides a multiplicative constant that is exponential in k.)

Before plunging into the proof of Theorem 18, let us explain why it implies the desired conclusion about quantum algorithms. The key observation relating quantum query complexity to low-degree polynomials was made by Beals et al. [5] in 1998:

Lemma 19 (Beals et al. [5]) *Given any quantum algorithm $\mathcal{A}$ that makes t queries to a Boolean input $x \in \{-1,1\}^N$, the probability that $\mathcal{A}$ accepts can be expressed as a real multilinear polynomial $p(x)$, of degree at most $2t$. (Thus, in particular, $p(x) \in [0,1]$ for all $x \in \{-1,1\}^N$.)*

Note that, *if* Theorem 18 worked for arbitrary polynomials (rather than only block-multilinear ones), then combining it with Lemma 19 would immediately give the simulation of quantum algorithms that we want.

Fortunately, one can strengthen Lemma 19, to show that a t-query quantum algorithm gives rise, not just to *any* bounded degree-$2t$ polynomial, but to a block-multilinear one.

Lemma 20 *Let $\mathcal{A}$ be a quantum algorithm that makes t queries to a Boolean input $x \in \{-1,1\}^N$. Then there exists a degree-$2t$ block-multilinear polynomial $p : \mathbb{R}^{2tN} \to \mathbb{R}$, with $2t$ blocks of N variables each, such that*

(i) the probability that $\mathcal{A}$ accepts x equals $p(x, \ldots, x)$ (with x repeated $2t$ times), and

(ii) $p(z) \in [-1,1]$ for all $z \in \{-1,1\}^{2tN}$.

Proof. Assume for simplicity (and without loss of generality) that $\mathcal{A}$ involves real amplitudes only.

For all $j \in [t]$ and $i \in [N]$, let $x_{j,i}$ be the value of x_i that $\mathcal{A}$'s oracle returns in response to its j^{th} query. Of course, in any "normal" run of $\mathcal{A}$, we will have $x_{j,i} = x_{j',i}$ for all j, j': that is, the value of x_i will be consistent across all t queries. But it is perfectly legitimate to ask what happens if x changes from one query to the next. In any case, $\mathcal{A}$ will have some normalized final state, of the form

$$\sum_{i,w} \alpha_{i,w}(x_{1,1}, \ldots, x_{t,N}) |i, w\rangle .$$

Furthermore, following Beals et al. [5], it is easy to see that each amplitude $\alpha_{i,w}$ can be written as a degree-t block-multilinear polynomial in the tN variables $x_{1,1}, \ldots, x_{t,N}$, with one block of N variables, $R_j = \{x_{j,1}, \ldots, x_{j,N}\}$, corresponding to each of the t queries. (If $\mathcal{A}$ has basis states that do not participate in queries, then we can deal with that by introducing dummy variables, $x_{1,0}, \ldots, x_{t,0}$, which are set to 1 in any "normal" run of $\mathcal{A}$.)

Next, for all $j \in [t]$ and $i \in [N]$, we create a *second* variable $x_{t+j,i}$, which just like $x_{j,i}$, represents the value of x_i that $\mathcal{A}$'s oracle returns in response to its j^{th} query. Let Acc be the set of all accepting basis states, and consider the polynomial

$$p\left(x_{1,1}, \ldots, x_{2t,N}\right) := \sum_{(i,w) \in \mathrm{Acc}} \alpha_{i,w}\left(x_{1,1}, \ldots, x_{t,N}\right) \alpha_{i,w}\left(x_{t+1,1}, \ldots, x_{2t,N}\right).$$

By construction, p is a degree-$2t$ block-multilinear polynomial in the $2tN$ variables $x_{1,1}, \ldots, x_{2t,N}$, with one block of N variables, $R_j = \{x_{j,1}, \ldots, x_{j,N}\}$, for each $j \in [2t]$. Furthermore, if we repeat the same input $x \in \{-1,1\}^N$ across all $2t$ blocks, then

$$p\left(x, \ldots, x\right) = \sum_{(i,w) \in \mathrm{Acc}} \alpha_{i,w}^2\left(x, \ldots, x\right)$$

is simply the probability that $\mathcal{A}$ accepts x. Finally, even if $x_{1,1}, \ldots, x_{2t,N} \in \{-1,1\}^{2tN}$ is completely arbitrary, $p\left(x_{1,1}, \ldots, x_{2t,N}\right)$ still represents an inner product between two vectors,

$$\sum_{(i,w) \in \mathrm{Acc}} \alpha_{i,w}\left(x_{1,1}, \ldots, x_{t,N}\right) |i,w\rangle \qquad \text{and} \qquad \sum_{(i,w) \in \mathrm{Acc}} \alpha_{i,w}\left(x_{t+1,1}, \ldots, x_{2t,N}\right) |i,w\rangle .$$

Since both of these vectors have norm at most 1, their inner product is bounded in $[-1,1]$. ■

As a side note, given any Boolean function $f : \{-1,1\}^N \to \{0,1\}$, one can consider the minimum degree of any block-multilinear polynomial p that approximates f. More formally, let the *block-multilinear approximate degree* of f, or $\widetilde{\operatorname{bmdeg}}(f)$, be the minimum degree of any block-multilinear polynomial $p : \mathbb{R}^{kN} \to \mathbb{R}$, with k blocks of N variables each, such that

(i) $|p(x, \ldots, x) - f(x)| \leq \frac{1}{3}$ for all $x \in \{-1,1\}^N$ (or alternatively, for all x satisfying some promise), and

(ii) $p\left(x_{1,1}, \ldots, x_{k,N}\right) \in [-1,1]$ for all $x_{1,1}, \ldots, x_{k,N} \in \{-1,1\}^{kN}$.

Recall that $\widetilde{\deg}(f)$, the "ordinary" approximate degree of f, is the minimum degree of any polynomial $p : \mathbb{R}^N \to \mathbb{R}$ such that $|p(x) - f(x)| \leq \frac{1}{3}$ for all x. Lemma 19 of Beals et al. [5] implies that $\widetilde{\deg}(f) \leq 2\,\mathrm{Q}(f)$ for all f, where $\mathrm{Q}(f)$ is the bounded-error quantum query complexity of f.

Clearly $\widetilde{\deg}(f) \leq \widetilde{\operatorname{bmdeg}}(f)$ for all f, by identifying variables across the k blocks. Also, Lemma 20 implies that $\widetilde{\operatorname{bmdeg}}(f) \leq 2\,\mathrm{Q}(f)$. Putting these facts together, we find that $\widetilde{\operatorname{bmdeg}}(f)$ is a lower bound on quantum query complexity that is *at least* as good as $\widetilde{\deg}(f)$, and *might* sometimes be better. We do not currently know whether there is any asymptotic separation between $\widetilde{\deg}(f)$ and $\widetilde{\operatorname{bmdeg}}(f)$, nor do we know whether there is an asymptotic separation between $\widetilde{\operatorname{bmdeg}}(f)$ and $\mathrm{Q}(f)$. Note that Ambainis [4] exhibited a Boolean function f such that $\widetilde{\deg}(f) = O\left(\mathrm{Q}(f)^{0.76}\right)$. By contrast, we do not know any techniques for upper-bounding $\widetilde{\operatorname{bmdeg}}(f)$, that do not *also* upper-bound $\mathrm{Q}(f)$.

5.1 Preprocessing the Polynomial

We are now ready to prove Theorem 18. Thus, suppose

$$p\left(x_{1,1},\ldots,x_{k,N}\right)=\sum_{i_1,\ldots,i_k\in[N]} a_{i_1,\ldots,i_k} x_{1,i_1}\cdots x_{k,i_k}$$

is a bounded block-multilinear polynomial of degree k. Then in our estimation procedure, the first step is to preprocess p, in order to "balance" it, and get rid of any variables that are "too influential." More formally, set $\delta := \varepsilon^2/N$. Then we wish to achieve the following requirement: for every nonempty set $S\subseteq[k]$,

$$\Lambda_S := \sum_{(i_j)_{j\in S}} \left(\sum_{(i_j)_{j\notin S}} a_{i_1,\ldots,i_k}\right)^2 \le \delta. \tag{5}$$

The basic operation that we use to achieve this requirement is *variable-splitting.* The operation consists of taking a variable $x_{j,l}$ and replacing it by m variables, in the following way. We introduce m new variables $x_{j,l_1},\ldots,x_{j,l_m}$, and define p' as the polynomial obtained by substituting $\frac{x_{j,l_1}+\cdots+x_{j,l_m}}{m}$ in the polynomial p instead of $x_{j,l}$. We refer to this as splitting $x_{j,l}$ into m variables. Observe that variable-splitting preserves the property that p is bounded in $[-1,1]$ at all Boolean points—for, regardless of how we set $x_{j,l_1},\ldots,x_{j,l_m}$, the value of p' will simply equal the value of p with $x_{j,l}$ set to $\frac{x_{j,l_1}+\cdots+x_{j,l_m}}{m}$, which in turn is a convex combination of p with $x_{j,l}$ set to -1 and p with $x_{j,l}$ set to 1.

Lemma 21 *Let $S\subseteq[k]$ be nonempty. Then there is a sequence of variable-splittings that introduces at most $1/\delta$ new variables, and that produces a polynomial p' that satisfies $\Lambda_S\le\delta$.*

Proof. We start with the case $S=[k]$. Then we have to ensure

$$\sum_{i_1,\ldots,i_k\in[N]} a^2_{i_1,\ldots,i_k} \le \delta, \tag{6}$$

where $a^2_{i_1,\ldots,i_k}$ is the coefficient of $x_{1,i_1}\ldots x_{k,i_k}$. Let

$$V_i := \sum_{i_2,\ldots,i_k\in[N]} a^2_{i_1,i_2,\ldots,i_k}.$$

We now randomly set each x_{j,i_j} for $j\ge 2$, to be 1 or -1 with independent probability 1/2. Let

$$X_i := \sum_{i_2,\ldots,i_k\in[N]} a_{i_1,i_2\ldots,i_k} x_{2,i_2}\cdots x_{k,i_k}.$$

Then $\mathrm{E}[X_i^2]=V_i$. By the concavity of the square root function, this means $\mathrm{E}[|X_i|]\ge\sqrt{V_i}$. Hence

$$E[|X_1|+\cdots+|X_N|]\ge\sqrt{V_1}+\cdots+\sqrt{V_N}.$$

If we set $x_{1,i} = 1$ whenever $X_i \geq 0$ and $x_{1,i} = -1$ otherwise, we get

$$p(x_{1,1}, \ldots, x_{k,N}) = \sum_{i=1}^{N} x_{1,i} X_i = \sum_{i=1}^{N} |X_i| \, .$$

Since $p(x_{1,1}, \ldots, x_{k,N})$ is bounded in $[-1, 1]$ at all Boolean points, this means that

$$\sqrt{V_1} + \cdots + \sqrt{V_N} \leq 1.$$

We now perform a sequence of variable-splittings. For each $i \in [N]$, let $m_i := \left\lfloor \sqrt{V_i}/\delta \right\rfloor$, so that

$$\delta m_i \leq \sqrt{V_i} < \delta \left(m_i + 1 \right).$$

Then we split $x_{1,i}$ into $m_i + 1$ variables. This replaces each term $a_{i_1,\ldots,i_k} x_{1,i_1} \cdots x_{k,i_k}$ with $m_i + 1$ terms that each equal $\frac{1}{m_i+1} a_{i_1,\ldots,i_k} x_{1,i_1} \cdots x_{k,i_k}$. Therefore, this variable-splitting reduces V_i to $V_i / \left(m_i + 1 \right)$, and decreases the sum (6) by $\frac{m_i}{m_i+1} V_i$.

After we have performed such variable-splittings for each i, the sum (6) becomes

$$\begin{aligned} \sum_{i=1}^{N} \frac{V_i}{m_i + 1} &\leq \sum_{i=1}^{N} \frac{V_i}{\sqrt{V_i}/\delta} \\ &= \delta \left(\sqrt{V_1} + \cdots + \sqrt{V_N} \right) \\ &\leq \delta. \end{aligned}$$

The number of new variables that get introduced equals

$$\sum_{i=1}^{N} m_i \leq \sum_{i=1}^{N} \frac{\sqrt{V_i}}{\delta} \leq \frac{1}{\delta}.$$

The case $S \subset [k]$ reduces to the case $S = [k]$ in the following way. For typographical convenience, assume that $S = [\ell]$ for some ℓ. Then substituting $x_{i,j} = 1$ for $i > \ell$ transforms the polynomial $p(x_{1,1}, \ldots, x_{k,N})$ into the polynomial

$$p'(x_{1,1}, \ldots, x_{\ell,N}) = \sum_{i_1,\ldots,i_\ell \in [N]} \bar{a}_{i_1,\ldots,i_\ell} x_{1,i_1} \cdots x_{\ell,i_\ell}$$

where

$$\bar{a}_{i_1,\ldots,i_\ell} := \sum_{i_{\ell+1},\ldots,i_k \in [N]} a_{i_1,\ldots,i_k}.$$

The statement of Lemma 21 now becomes

$$\sum_{i_1,\ldots,i_\ell \in [N]} \bar{a}^2_{i_1,\ldots,i_\ell} \leq \delta$$

which can be achieved similarly to the previous case. ■

Lemma 21 has the following consequence.

Corollary 22 *There is a sequence of variable-splittings that introduces at most $2^k/\delta$ new variables, and that produces a polynomial p' that satisfies $\Lambda_S \leq \delta$ for every nonempty subset $S \subseteq [k]$.*

Proof. We simply apply the procedure of Lemma 21 once for each nonempty $S \subseteq [k]$, in any order. Since there are $2^k - 1$ possible choices for S, and since each iteration adds at most $1/\delta$ variables, the total number of added variables is at most $2^k/\delta$. Furthermore, we claim that later iterations can never "undo" the effects of previous iterations. This is because, if we consider how the quantity

$$\Lambda_S = \sum_{(i_j)_{j \in S}} \left(\sum_{(i_j)_{j \notin S}} a_{i_1,\ldots,i_k} \right)^2$$

is affected by variable-splittings applied to the variables in R_j, there are only two possibilities: if $j \in S$ then Λ_S can decrease, while if $j \notin S$ then Λ_S remains unchanged. ■

We now apply Corollary 22 with the choice $\delta = \varepsilon^2/N$. This introduces at most $2^k N/\varepsilon^2 = O\left(N/\varepsilon^2\right)$ new variables, and achieves $\Lambda_S \leq \varepsilon^2/N$ for every S.

From now on, we will use n to denote the "new" number of variables per block, which is a constant factor greater than the "old" number N.

5.2 The Estimator

Let

$$b_{i_1,\ldots,i_k} := a_{i_1,\ldots,i_k} x_{1,i_1} \cdots x_{k,i_k}.$$

Then

$$p(x_{1,1},\ldots,x_{k,n}) = \sum_{i_1,\ldots,i_k} b_{i_1,\ldots,i_k}.$$

We can estimate this sum in the following way. For each i, j_i independently, let y_{i,j_i} be a $\{0,1\}$-valued random variable with $\Pr[y_{i,j_i} = 1] = \frac{1}{n^{1/k}}$. We then take

$$P := b_{i_1,\ldots,i_k} y_{1,i_1} \cdots y_{k,i_k}$$

as our estimator.

Clearly, this is an unbiased estimator of $p(x_{1,1},\ldots,x_{k,n})$, with expectation

$$\mathrm{E}[P] = \frac{p(x_1,\ldots,x_n)}{n}.$$

The result we would like to prove is that $\mathrm{Var}[P] = O(\delta/n)$. If this is true, then performing $O(1)$ repetitions of P allows us to estimate $p(x_{1,1},\ldots,x_{k,n})$ with precision $\sqrt{\delta n} = \sqrt{(\varepsilon^2/n) \cdot n} = \varepsilon$. This estimation can be carried out with $O(n^{1-1/k})$ queries because, to calculate P, we only need the values of $x_{i,j}$ with $y_{i,j} = 1$, and the number of such variables is $O(n^{1-1/k})$, with a very high probability. Note that

$$O(n^{1-1/k}) = O\left(\left(\frac{N}{\delta}\right)^{1-1/k}\right) = O\left(\left(\frac{N}{\varepsilon^2}\right)^{1-1/k}\right),$$

in terms of the number of variables N of our original polynomial. Here the big-O hides a factor of $\exp(k)$.

5.3 Warmup

As a warmup, consider the following simpler estimator. For each $i_1, \ldots, i_k$ independently, let $y_{i_1,\ldots,i_k}$ be a $\{0,1\}$-valued random variable with

$$\Pr[y_{i_1,\ldots,i_k} = 1] = \frac{1}{n}.$$

Then let

$$P' := \sum_{i_1,\ldots,i_k \in [n]} b_{i_1,\ldots,i_k} y_{i_1,\ldots,i_k}.$$

Once again, P' is clearly an unbiased estimator for p, with expectation $\mathrm{E}[P'] = p/n$.

Let us start by proving that $\mathrm{Var}[P'] = O(\delta/n)$. Let

$$B = \sum_{i_1,\ldots,i_k \in [n]} b_{i_1,\ldots,i_k}^2.$$

Then

$$\begin{aligned}
\mathrm{Var}[P'] &= \sum_{i_1,\ldots,i_k \in [n]} b_{i_1,\ldots,i_k}^2 \,\mathrm{Var}[y_{i_1,\ldots,i_k}] \\
&= \sum_{i_1,\ldots,i_k \in [n]} b_{i_1,\ldots,i_k}^2 \left(\frac{1}{n} - \frac{1}{n^2}\right) \\
&\leq \frac{1}{n} \sum_{i_1,\ldots,i_k \in [n]} b_{i_1,\ldots,i_k}^2 \\
&= \frac{B}{n}.
\end{aligned}$$

Taking $S = [k]$ in equation (5) implies that $B \leq \delta$ and hence $\mathrm{Var}[P'] \leq \delta/n$.

5.4 Second Estimator

The variance of the original estimator P is

$$\begin{aligned}
\mathrm{Var}[P] = &\sum_{i_1,\ldots,i_k \in [n]} b_{i_1,\ldots,i_k}^2 \,\mathrm{Var}\left[y_{1,i_1} \cdots y_{k,i_k}\right] \\
&+ \sum_{(i_1,\ldots,i_k) \neq (i'_1,\ldots,i'_k)} b_{i_1,\ldots,i_k} b_{i'_1,\ldots,i'_k} \,\mathrm{Cov}\left[y_{1,i_1} \cdots y_{k,i_k}, y_{1,i'_1} \cdots y_{k,i'_k}\right] \\
&= \mathrm{Var}[P'] + \sum_{(i_1,\ldots,i_k) \neq (i'_1,\ldots,i'_k)} b_{i_1,\ldots,i_k} b_{i'_1,\ldots,i'_k} \,\mathrm{Cov}\left[y_{1,i_1} \cdots y_{k,i_k}, y_{1,i'_1} \cdots y_{k,i'_k}\right].
\end{aligned}$$

If $i_j \neq i'_j$ for all j, then $\prod_j y_{j,i_j}$ and $\prod_j y_{j,i'_j}$ are independent random variables and the covariance between them is zero. If $i_j = i'_j$ for ℓ values of j, then

$$\begin{aligned}\mathrm{Cov}\left[\prod_j y_{j,i_j}, \prod_j y_{j,i'_j}\right] &= \Pr\left[\prod_j y_{j,i_j} = \prod_j y_{j,i'_j} = 1\right] - \Pr\left[\prod_j y_{j,i_j} = 1\right] \Pr\left[\prod_j y_{j,i'_j} = 1\right] \\ &= \frac{1}{\left(n^{1/k}\right)^{2k-\ell}} - \left(\frac{1}{\left(n^{1/k}\right)^{k}}\right)^2 \\ &= \frac{1}{n^{2-\ell/k}} - \frac{1}{n^2}.\end{aligned}$$

Let S_ℓ consist of all pairs $(i_1, \ldots, i_k), (i'_1, \ldots, i'_k)$ such that $i_j = i'_j$ for exactly ℓ values of j. Let T_ℓ be the multiset consisting of the elements of $S_\ell, \ldots, S_{k-1}$, with each element of $S_{\ell'}$ occurring $\binom{\ell'}{\ell}$ times. Then by inclusion-exclusion, we have

$$S_\ell = T_\ell - \binom{\ell+1}{\ell} T_{\ell+1} + \binom{\ell+2}{\ell} T_{\ell+2} - \cdots$$

where $\binom{\ell'}{\ell} T_{\ell'}$ denotes the union of ℓ' copies of $T_{\ell'}$. Hence,

$$\begin{aligned}\mathrm{Var}[P] &= \mathrm{Var}[P'] + \sum_{\ell=1}^{k-1} \left(\frac{1}{n^{2-\ell/k}} - \frac{1}{n^2}\right) \sum_{(i_1,\ldots,i_k),(i'_1,\ldots,i'_k) \in S_\ell} b_{i_1,\ldots,i_k} b_{i'_1,\ldots,i'_k} \\ &= \mathrm{Var}[P'] + \sum_{\ell=1}^{k-1} p_\ell \sum_{(i_1,\ldots,i_k),(i'_1,\ldots,i'_k) \in T_\ell} b_{i_1,\ldots,i_k} b_{i'_1,\ldots,i'_k} \qquad (7)\end{aligned}$$

where

$$p_\ell := \sum_{j=1}^{\ell} (-1)^{\ell-j} \binom{\ell}{j} \left(\frac{1}{n^{2-j/k}} - \frac{1}{n^2}\right).$$

For large n, we have $p_\ell = (1 \pm o(1)) \frac{1}{n^{2-\ell/k}}$. To complete the proof, we just need one more lemma.

Lemma 23

$$\sum_{(i_1,\ldots,i_k),(i'_1,\ldots,i'_k) \in T_\ell} b_{i_1,\ldots,i_k} b_{i'_1,\ldots,i'_k} \leq \delta \binom{k}{\ell}.$$

Proof. Let $S \subseteq [k]$ with $|S| = \ell$. We define T_S as the set consisting of all pairs $(i_1, \ldots, i_k), (i'_1, \ldots, i'_k)$ such that $i_j = i'_j$ for all $j \in S$ and $(i_1, \ldots, i_k) \neq (i'_1, \ldots, i'_k)$. Then

$$T_\ell = \sum_{S \,:\, |S|=\ell} T_S,$$

and

$$\sum_{\left((i_1,\ldots,i_k),(i'_1,\ldots,i'_k)\right) \in T_\ell} b_{i_1,\ldots,i_k} b_{i'_1,\ldots,i'_k} = \sum_{S \,:\, |S|=\ell} \sum_{\left((i_1,\ldots,i_k),(i'_1,\ldots,i'_k)\right) \in T_S} b_{i_1,\ldots,i_k} b_{i'_1,\ldots,i'_k}.$$

The lemma now follows by showing that, for each S, the inner sum is at most δ. To show that, we first add all pairs $((i_1,\ldots,i_k),(i_1,\ldots,i_k))$ to T_S. This may only increase the sum because $a^2_{i_1,\ldots,i_k}$ is always at least 0. Then, we group together all terms with the same values of $i_j = i'_j$ for $j \in S$. The sum of those is equal to

$$\sum_{(i_j,i'_j)_{j\notin S}} b_{i_1,\ldots,i_k} b_{i'_1,\ldots,i'_k} = \left(\sum_{(i_j)_{j\notin S}} b_{i_1,\ldots,i_k} \right)^2 .$$

Because of (5), the sum of all such squares, over all $i_j, j \in S$ is at most δ. ■

Combining Lemma 23 with (7), we obtain

$$\begin{aligned} \mathrm{Var}[P] &= \mathrm{Var}[P'] + \sum_{\ell=1}^{k-1} p_\ell \cdot O(\delta) \\ &= \mathrm{Var}[P'] + \sum_{\ell=1}^{k-1} O\left(\frac{\delta}{n^{2-\ell/k}}\right) \\ &= \mathrm{Var}[P'] + O\left(\frac{\delta}{n^{1+1/k}}\right) \\ &= O\left(\frac{\delta}{n}\right). \end{aligned}$$

6 BQP-Completeness

In this section, we prove that (an explicit version of) the k-fold Forrelation problem, with $k = \mathrm{poly}(n)$, is complete for the complexity class PromiseBQP. More generally, for any k, we will show how explicit k-fold Forrelation captures the power of quantum circuits of depth $O(k)$.

Recall that, in explicit k-fold Forrelation, we are given as input k Boolean circuits $C_1,\ldots,C_k$, which compute the Boolean functions $f_1,\ldots,f_k : \{0,1\}^n \to \{-1,1\}$ respectively. The problem is to decide whether the "twisted sum"

$$\Phi_{f_1,\ldots,f_k} := \frac{1}{2^{(k+1)n/2}} \sum_{x_1,\ldots,x_k \in \{0,1\}^n} f_1(x_1)(-1)^{x_1\cdot x_2} f_2(x_2)(-1)^{x_2\cdot x_3} \cdots (-1)^{x_{k-1}\cdot x_k} f_k(x_k)$$

satisfies $|\Phi_{f_1,\ldots,f_k}| \le \frac{1}{100}$ or $\Phi_{f_1,\ldots,f_k} \ge \frac{3}{5}$, promised that one of those is the case. As we observed in Proposition 6, this problem is clearly *in* PromiseBQP, so our task reduces to showing that it's PromiseBQP-hard—i.e., that any quantum circuit can be encoded into it.

For this task, it will suffice to consider an extremely restricted version of k-fold Forrelation, in which each function f_i depends on at most 3 of its input bits.

We will appeal to a well-known result of Shi [21], who showed that the gate set $\{\mathrm{H}, \mathrm{Toffoli}\}$ is already universal for quantum computation. Recall here that

$$\mathrm{H} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

is the Hadamard gate, while the Toffoli gate is the 3-qubit gate that maps each basis vector $|x,y,z\rangle$ to $|x,y,z\oplus xy\rangle$. The Toffoli gate is equivalent, under conjugating the third qubit by Hadamards, to the controlled-controlled-sign or CCSIGN gate, which maps each $|x, y, z\rangle$ to $(-1)^{xyz} |x, y, z\rangle$. Thus, we deduce that the set $\{\mathrm{H}, \mathrm{CCSIGN}\}$ is also universal for quantum computation.

In a bit more detail, given a quantum circuit Q composed of Hadamard and CCSIGN gates, acting on n qubits, define

$$A_Q := \langle 0|^{\otimes n} Q |0\rangle^{\otimes n},$$

so that A_Q^2 is the probability that Q returns the all-0 state to itself. Then let QSim be the problem of deciding whether $|A_Q| \leq \frac{1}{100}$ or $A_Q \geq \frac{3}{5}$, promised that one of those is the case.

Lemma 24 (follows from Shi [21]) *QSim is PromiseBQP-complete.*

Proof. Besides what was said above, together with standard amplification, we just need two further observations. First, by using uncomputing, we can modify any quantum circuit so that it "accepts" by returning all its qubits to the initial state, $|0\rangle^{\otimes n}$, and "rejects" by ending in any state orthogonal to $|0\rangle^{\otimes n}$. Second, we can handle the case that A_Q is negative by running both Q and $-Q$ (i.e., Q with a -1 global phase), and checking whether our QSim oracle returns $A_Q \geq \frac{3}{5}$ for either of them. ■

Now, the outline of a reduction from QSim to k-fold Forrelation suggests itself almost immediately. Given an n-qubit quantum circuit Q over the basis $\{\mathrm{H}, \mathrm{CCSIGN}\}$, we want to construct Boolean functions $f_1, \ldots, f_k$ with the property that $\Phi_{f_1,\ldots,f_k} = A_Q$. To do so, we should exploit the fact that, as we have seen, $\Phi_{f_1,\ldots,f_k}$ is a transition amplitude for a particular kind of quantum circuit: namely, a circuit that consists of rounds of Hadamards applied to all n qubits, interleaved with diagonal matrices U_{f_i} that map each basis state $|x\rangle$ to $f_i(x)|x\rangle$. Thus, we should use suitably-placed f_i's to simulate each of the CCSIGN gates in Q (exploiting the fact that CCSIGN is diagonal in the computational basis), while using the $(-1)^{x_i \cdot x_{i+1}}$ terms in the expression for $\Phi_{f_1,\ldots,f_k}$ to simulate the Hadamard gates in Q.

However, there is a technical problem in implementing the above plan. Namely, while $\Phi_{f_1,\ldots,f_k}$ *will* equal the transition amplitude $\langle 0|^{\otimes n} Q' |0\rangle^{\otimes n}$, for some quantum circuit Q' that consists of Hadamard and CCSIGN gates, the circuit Q' will contain n Hadamard gates between every CCSIGN gate and the succeeding one, *whether we want Hadamards there or not.* This suggests that, in order to encode an *arbitrary* sequence of Hadamard and CCSIGN gates, we need some gadget that "cancels" unwanted Hadamard gates against each other, leaving only the Hadamard gates that actually appear in the original circuit Q. Of course, we can exploit the fact that H^2 is the identity. So for example, if we wanted to remove the n Hadamard gates that "automatically appear" between $U_{f_{i-1}}$ and U_{f_i}, then we could simply set f_i to be the constant 1 function, so that U_{f_i} was the identity. Then every H between $U_{f_{i-1}}$ and U_{f_i} would cancel with a corresponding H between U_{f_i} and $U_{f_{i+1}}$. Alas, this still doesn't tell us how to cancel *some* H's: that is, how to Hadamard certain desired qubits, but not other qubits. We do this in the following theorem.

Theorem 25 *Explicit k-fold* Forrelation, *for $k = \mathrm{poly}(n)$, is* PromiseBQP-*hard. (Moreover, the functions $f_1, \ldots, f_k$ produced by the reduction all have the form $f_i(x) = (-1)^{C(x)}$, where C is a product of at most* 3 *input bits.)*

Proof. Given what we said above, the only additional ingredient we need is a gadget that lets us Hadamard some desired *subset* of the qubits, $S \subset [n]$, and not the qubits outside S.

For simplicity, suppose that $|S| = 2$, and let a and b be S's elements. Our gadget, shown in Figure 2, consists of three CSIGN gates (i.e., gates that map $|x, y\rangle$ to $(-1)^{xy} |x, y\rangle$) on a and

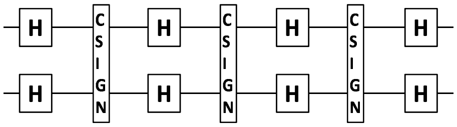

Figure 2: A 2-qubit gadget for converting an even number of layers of Hadamard gates into an odd number.

b, sandwiched between Hadamard gates. Note that we can implement a CSIGN on a and b as U_{f_i}, where $f_i(z_1, \ldots, z_n) := (-1)^{z_a z_b}$. Meanwhile, the Hadamard gates are just those that are automatically applied between each U_{f_i} and $U_{f_{i+1}}$ in a quantum circuit for Forrelation.

To see why the gadget works, consider the following identity:

$$\left(\frac{1}{2}\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix}\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}\right)^3 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

In particular, if we let C stand for CSIGN, $\mathrm{H}^{\otimes 2}$ for Hadamards on two qubits, and S for the 2-qubit SWAP gate, then

$$\mathrm{H}^{\otimes 2}\,\mathrm{C}\,\mathrm{H}^{\otimes 2}\,\mathrm{C}\,\mathrm{H}^{\otimes 2}\,\mathrm{C}\,\mathrm{H}^{\otimes 2} = \mathrm{S}\,\mathrm{H}^{\otimes 2}.$$

Contrast this with what happens if we apply the 2-qubit identity, I, rather than C, in the inner layers:

$$\mathrm{H}^{\otimes 2}\,\mathrm{I}\,\mathrm{H}^{\otimes 2}\,\mathrm{I}\,\mathrm{H}^{\otimes 2}\,\mathrm{I}\,\mathrm{H}^{\otimes 2} = \mathrm{I}.$$

Thus, Hadamards get applied if C is chosen for the inner layers, but *not* if I is chosen. So this gadget has the effect of Hadamarding a and b, while not Hadamarding the other qubits in the circuit. Now, the gadget also has the unintended side effect of swapping a and b. But since we know this is going to happen, we can keep track of it by simply swapping the *labels* of a and b whenever the gadget is applied.

To generalize to arbitrary subsets $S \subset [n]$: if $|S| > 2$ is even, then we simply partition S into pairs, and apply the 2-qubit Hadamard gadget once in succession to each pair. If $|S|$ is odd, then the odd qubit in S can be paired with a "dummy qubit," which is introduced into the circuit for this sole purpose.

Notice that each CSIGN gate is simulated by a single f_i, while each pair of Hadamards is simulated by three f_i's together with the Hadamards that sandwich them. Thus, we can place a pair of Hadamards after a CSIGN gate, or vice versa, with no difficulty. To place one CSIGN gate after another, or one pair of Hadamards after another, we insert an $f_i = 1$ (i.e., a constant 1 function) in between them, in order to cancel the unwanted Hadamards.

Given a quantum circuit Q on n qubits, consisting of m Hadamard and CCSIGN gates, the end result of our reduction will be a list of Boolean functions $f_1, \ldots, f_k : \{0,1\}^{n+1} \to \{-1,1\}$, with $k = O(m)$, such that $\Phi_{f_1,\ldots,f_k} = A_Q$. (The $n+1$ comes from the addition of the dummy qubit.) Furthermore, each $f_i(z_1, \ldots, z_n)$ in the list will have the form 1 or $(-1)^{z_a z_b}$ or $(-1)^{z_a z_b z_c}$, so will be easy to specify using a Boolean circuit. ■

As a side note, suppose we required the functions $f_1, \ldots, f_k$ to depend on at most 2 input bits, rather than 3 bits. In that case, we claim that k-fold Forrelation would be in P. The reason is just that in this case, our quantum circuit for FORRELATION would be a stabilizer circuit, so the Gottesman-Knill Theorem would apply.[16]

Examining the proof of Theorem 25, we can derive a stronger consequence. Define a *depth-d quantum circuit* as one where the gates are organized into d sequential layers, with the gates within each layer all commuting with one another.[17] Now, given a depth-d quantum circuit Q over the basis $\{\mathrm{H}, \mathrm{CCSIGN}\}$, let QSIM_d be the problem of deciding whether $A_Q := \langle 0|^{\otimes n} Q |0\rangle^{\otimes n}$ satisfies $A_Q \geq \frac{1}{4}$ or $|A_Q| \leq \frac{1}{100}$, promised that one of those is the case. Then we have the following:

Theorem 26 QSIM_d *is polynomial-time reducible to explicit* $(2d+1)$*-fold* FORRELATION. *(Moreover, the functions $f_1, \ldots, f_{2d+1}$ produced by the reduction all have the form $f_i(x) = (-1)^{p(x)}$, where p is a degree-*3 *polynomial in the input bits.)*

Proof. The only change we need to make to the proof of Theorem 25 is to be a bit more frugal with f_i's—using at most two f_i's for each layer of Q, rather than separate f_i's for each gate.

In more detail, a given layer L of Q consists of Hadamard gates on some subset of qubits $S \subseteq [n]$, as well as CCSIGN gates (which might overlap each other) on some other subset of qubits $T \subseteq [n]$ satisfying $S \cap T = \varnothing$. Suppose we want to simulate L using the three functions f_i, f_{i+1}, f_{i+2}, together with the Hadamards that sandwich them. Then we build up the functions as follows: initially $f_i = f_{i+1} = f_{i+2} = 1$. For each CCSIGN gate, acting on qubits $a, b, c \in T$, we multiply f_{i+1} by $(-1)^{z_a z_b z_c}$, leaving f_i and f_{i+2} unchanged. For each pair of Hadamard gates, acting on qubits $a, b \in S$, we multiply f_i, f_{i+1}, and f_{i+2} by $(-1)^{z_a z_b}$. One can check that the end result is

$$\mathrm{H}^{\otimes n} U_{f_{i+2}} \mathrm{H}^{\otimes n} U_{f_{i+1}} \mathrm{H}^{\otimes n} U_{f_i} \mathrm{H}^{\otimes n} = \sigma L,$$

where σ represents a SWAP gate applied to each pair $a, b \in S$ (something that, as before, we can easily keep track of).

To separate two successive layers of the circuit, we could simply insert a constant function, $f_i = 1$. This would yield a $4d$-fold FORRELATION instance, d being the number of layers. If we want to decrease the number of f_i's from $4d$ to $2d+1$, then we can eliminate each constant f_i, together with the Hadamard layers surrounding it (which simply cancel each other out), and then merge f_{i-1} and f_{i+1} into a single f_i by multiplying them: $f_i(x) = f_{i-1}(x) f_{i+1}(x)$, or $f_i(x) = (-1)^{p(x)+q(x)}$ if $f_{i-1}(x) = (-1)^{p(x)}$ and $f_{i+1}(x) = (-1)^{q(x)}$.

Note that, at the end, each f_i is a degree-3 polynomial in its input bits, and we again have $\Phi_{f_1,\ldots,f_k} = A_Q$. ■

So for example, we find that explicit $\log n$-fold FORRELATION is a complete promise problem for $\mathsf{PromiseBQNC}^1$: the class of problems that captures what can be done using log-depth quantum circuits (and which already contains FACTORING, by a result of Cleve and Watrous [13]).

7 Appendix: Separations for Sampling and Relation Problems

Let FOURIER SAMPLING be the following problem. Given oracle access to a Boolean function $f : \{0,1\}^n \to \{-1,1\}$, the task is to sample from a distribution D over $\{0,1\}^n$ such that $\|D - D_f\| \leq \varepsilon$,

[16] Indeed, by a result of Aaronson and Gottesman [3], k-fold FORRELATION with this restriction is $\oplus\mathsf{L}$-complete.

[17] Often, one further requires that the gates within each layer act on disjoint sets of qubits. But it will be convenient for us to drop that requirement.

where D_f is the distribution defined by

$$\Pr_{D_f}[y] = \hat{f}(y)^2 = \left(\frac{1}{2^n} \sum_{x \in \{0,1\}^n} f(x)(-1)^{x \cdot y} \right)^2 .$$

It is clear that Fourier Sampling is solvable—indeed, with $\varepsilon = 0$—by a quantum algorithm that makes just a single query to f. The algorithm consists of a round of Hadamard gates, then a query to f, then another round of Hadamard gates, then a measurement in the computational basis.

By contrast, we show in this appendix that any classical randomized algorithm for Fourier Sampling requires $\Omega(N/\log N)$ queries, where $N = 2^n$ is the size of f's truth table. In other words, a much larger quantum versus classical separation can be achieved for sampling problems than for decision problems.

Theorem 27 *Fix (say) $\varepsilon = 0.01$. Then the randomized query complexity of* Fourier Sampling *is $\Omega(N/\log N)$.*

Proof. Let A be a classical algorithm, and let $\mathcal{E}_f = \{p_{f,y}\}_{y \in \{0,1\}^n}$ be the probability distribution output by A when given f as an oracle. The success condition is that, for all f,

$$\frac{1}{2} \sum_{y \in \{0,1\}^n} \left| p_{f,y} - \hat{f}(y)^2 \right| \leq \varepsilon.$$

By an averaging argument, this implies that there exists a $y^* \in \{0,1\}^n$ such that

$$\operatorname*{E}_f \left[\left| p_{f,y^*} - \hat{f}(y^*)^2 \right| \right] \leq \frac{2\varepsilon}{N}.$$

So by Markov's inequality,

$$\left| p_{f,y^*} - \hat{f}(y^*)^2 \right| \leq \frac{20\varepsilon}{N}$$

for at least a $9/10$ fraction of f's. Now assume by symmetry, and without loss of generality, that $y^* = 0^n$. Let $z_i := \frac{1+f(x_i)}{2}$ (where $x_1, \ldots, x_N$ is a lexicographic ordering of inputs), let $Z := (z_1, \ldots, z_N)$, and let

$$|Z| := z_1 + \cdots + z_N.$$

Then $\hat{f}(0^n) = (2|Z| - N)/N$. The question before us is how many z_i's the algorithm A needs to query, in order to output 0^n (or, as we'll say, "accept") with a probability $p_Z := p_{f,0^n}$ that satisfies

$$\left| p_Z - \left(\frac{2|Z|}{N} - 1 \right)^2 \right| \leq \frac{20\varepsilon}{N} \tag{8}$$

with probability at least $9/10$ over $Z \in \{0,1\}^N$.

Observe that, without loss of generality, A just nonadaptively queries t randomly-chosen inputs $z_{i_1}, \ldots, z_{i_t}$, and then accepts with a probability q_k that depends solely on $k := z_{i_1} + \cdots + z_{i_t}$. For, if A did anything other than this, then by averaging over all $N!$ possible permutations of Z, we would obtain an algorithm of this restricted form that made the same number of queries and that was just as likely to satisfy (8). In particular, this means that the probability $p_{|Z|} = p_Z$ that A accepts Z depends only on $|Z|$. Explicitly,

$$p_w = \sum_{k=0}^{t} q_k r_{k,w}, \tag{9}$$

where

$$r_{k,w} = \frac{1}{2^N} \binom{t}{k} \binom{N-t}{w-k}$$

is the probability that $z_{i_1} + \cdots + z_{i_t} = k$ conditioned on $|Z| = w$.

Let

$$\begin{aligned} U &:= \left\{ Z : \left| |Z| - \frac{N}{2} \right| \leq \frac{\sqrt{N}}{4} \right\}, \\ V &:= \left\{ Z : \left| |Z| - \frac{N}{2} \right| \in \left[\frac{\sqrt{N}}{2}, 2\sqrt{N} \right] \right\}. \end{aligned}$$

Then note that for sufficiently large N,

$$\begin{aligned} \Pr_Z \left[Z \in U \right] &\geq \operatorname{erf}\left(\frac{1}{2\sqrt{2}} \right) - o\left(1\right) > 0.38, \\ \Pr_Z \left[Z \in V \right] &\geq \operatorname{erf}\left(2\sqrt{2} \right) - \operatorname{erf}\left(\frac{1}{\sqrt{2}} \right) - o\left(1\right) > 0.31. \end{aligned}$$

This implies that, conditioned on $Z \in U$, we must have

$$\begin{aligned} p_Z &\leq \left(\frac{2\left|Z\right|}{N} - 1 \right)^2 + \frac{20\varepsilon}{N} \\ &\leq \left(\frac{2\sqrt{N}/4}{N} \right)^2 + \frac{20\varepsilon}{N} \\ &= \frac{0.25 + 20\varepsilon}{N} \\ &= \frac{0.45}{N} \end{aligned}$$

with probability at least $1 - \frac{0.1}{0.38} > 2/3$ over Z. Likewise, conditioned on $Z \in V$, we must have

$$\begin{aligned} p_Z &\geq \left(\frac{2\left|Z\right|}{N} - 1 \right)^2 - \frac{20\varepsilon}{N} \\ &\geq \frac{1 - 20\varepsilon}{N} \\ &= \frac{0.8}{N} \end{aligned}$$

with probability at least $1 - \frac{0.1}{0.31} > 2/3$ over Z.

Thus, it suffices to prove a lower bound for the following restricted problem: for at least $2/3$ of strings $Z \in U$, accept with some probability $p_Z \leq 0.45/N$, while for at least $2/3$ of strings $Z \in V$, accept with some probability $p_Z \geq 0.8/N$. Indeed, let us assume without loss of generality that if Z is drawn from U then $|Z| = N/2$ exactly, while if Z is drawn from V then $|Z| = N/2 + 2\sqrt{N}$ exactly. This can only make the distinguishing task easier, and therefore the lower bound stronger. Note that, because $p_{|Z|} = p_Z$ depends only on $|Z|$, any algorithm that achieves $p_Z \leq 0.45/N$ for at least a $2/3$ fraction of $|Z| = N/2$ actually achieves that for *all* $|Z| = N/2$, while any algorithm that achieves $p_Z \geq 0.8/N$ for at least a $2/3$ fraction of $|Z| = N/2 + 2\sqrt{N}$ achieves that for all $|Z| = N/2 + 2\sqrt{N}$.

Recall from (9) that $p_{|Z|} = p_Z$ is a linear combination of $r_{k,|Z|}$'s, which are the probabilities for various numbers k of '1' bits to be observed among the t bits queried, conditioned on $|Z|$ having the value that it does. Moreover, the coefficients q_k in this linear combination are all in $[0,1]$. We want to show that, if $t = o(N/\log N)$, then either $p_{N/2+2\sqrt{N}} = o(1/N)$ or else $p_{N/2+2\sqrt{N}} - p_{N/2} = o(1/N)$—either of which suffices to show A's failure.

We deduce this from two probabilistic claims. First, by a Chernoff bound,

$$\begin{aligned}
\sum_{k=t/2+c\sqrt{t}}^{t} r_{k,N/2+2\sqrt{N}} &= \Pr\left[z_{i_1} + \cdots + z_{i_t} \geq \frac{t}{2} + c\sqrt{t} : |Z| = \frac{N}{2} + 2\sqrt{N}\right] \\
&\leq \exp\left\{-\frac{1}{3}\frac{\left(\frac{t}{2} + c\sqrt{t} - \left(\frac{t}{2} + \frac{2t}{\sqrt{N}}\right)\right)^2}{\frac{t}{2} + \frac{2t}{\sqrt{N}}}\right\} \\
&\leq \exp\left\{-\frac{1}{3t}\left(c^2 t - \frac{4ct^{3/2}}{\sqrt{N}} + \frac{4t^2}{N}\right)\right\} \\
&\leq \exp\left\{-\frac{c^2}{3} + \frac{4c}{3}\sqrt{\frac{t}{N}}\right\} \\
&\leq \exp\left\{-\Omega\left(c^2\right)\right\}
\end{aligned}$$

for large c. So in particular, if $c = \omega(\sqrt{\log N})$, then all the events involving observing k '1' bits, for $k \geq t/2 + c\sqrt{t}$, have total probability $o(1/N)$. This means that, if A worked, then we could set $q_k = 0$ for all $k \geq t/2 + c\sqrt{t}$ without affecting A's success: we would still have $p_{N/2+2\sqrt{N}} - p_{N/2} = \Omega(1/N)$ and $p_{N/2} = O(1/N)$.

Thus, let us concentrate next on $r_{k,N/2+2\sqrt{N}}$ and $r_{k,N/2}$ for $k \leq t/2 + O(\sqrt{t \log N})$. Here, we look at their ratio:

$$\begin{aligned}
\frac{r_{k,N/2+2\sqrt{N}}}{r_{k,N/2}} &= \frac{\binom{N-t}{N/2+2\sqrt{N}-k}}{\binom{N-t}{N/2-k}} \\
&= \frac{\left(N/2-t+k-2\sqrt{N}+1\right)\cdots(N/2-t+k)}{(N/2-k+1)\cdots\left(N/2-k+2\sqrt{N}\right)} \\
&\leq \left(1+\frac{2k-t-2\sqrt{N}}{N/2-k+1}\right)^{2\sqrt{N}} \\
&\leq \left(1+\frac{2k-t}{N/4}\right)^{2\sqrt{N}} \\
&\leq \left(1+O\left(\frac{\sqrt{t}\log N}{N}\right)\right)^{2\sqrt{N}} \\
&= \exp\left\{O\left(\sqrt{\frac{t\log N}{N}}\right)\right\}.
\end{aligned}$$

Notice that, if $t = o(N/\log N)$, then the above ratio is $1 + o(1)$. This means that taking a nonnegative linear combination of $r_{k,|Z|}$'s cannot possibly suffice to achieve $p_{N/2} = O(1/N)$ and $p_{N/2+2\sqrt{N}} - p_{N/2} = \Omega(1/N)$ at the same time. ■

We conjecture that Theorem 27 is tight: that is, that there exists a randomized algorithm for Fourier Sampling making $O(N/\log N)$ queries. More generally, we conjecture that *any* approximate sampling problem solvable with 1 quantum query is also solvable with $O(N/\log N)$ classical randomized queries. Still more generally, we conjecture that any approximate sampling problem solvable with $k = O(1)$ quantum queries is also solvable with $O(N/(\log N)^{1/k})$ classical randomized queries; and that this is tight, being achieved by a k-fold generalization of Fourier Sampling. In the k-fold generalization, we are given oracle access to k Boolean functions $f_1, \ldots, f_k : \{0,1\}^n \to \{-1,1\}$. The task is to sample from a distribution D over $\{0,1\}^n$ such that $\|D - D_{f_1,\ldots,f_k}\| \leq \varepsilon$, where $D_{f_1,\ldots,f_k}$ is the distribution defined by

$$\Pr_{D_{f_1,\ldots,f_k}}[y] = \left(\frac{1}{2^{n(k+1)/2}}\sum_{x_1,\ldots,x_k\in\{0,1\}^n} f_1(x_1)(-1)^{x_1\cdot x_2} f_2(x_2)(-1)^{x_2\cdot x_3}\cdots f_k(x_k)(-1)^{x_k\cdot y}\right)^2.$$

So far, we have discussed separations for approximate sampling problems. But it is also possible to modify Fourier Sampling to produce a *relation* problem—that is, a problem of outputting any element of a set S of "valid solutions"—with a large quantum/classical separation. One way to do this would be to use the construction of Aaronson [2], which, given any approximate sampling problem, uses Kolmogorov complexity to produce a relation problem of roughly equivalent difficulty. Unfortunately, that construction will blow up the quantum query complexity from 1 to $O(\log N)$, weakening the result. A more direct approach would be to consider the following relation problem: given oracle access to a Boolean function $f : \{0,1\}^n \to \{-1,1\}$, output any string $y \in \{0,1\}^n$ such that $\left|\hat{f}(y)\right| \geq c$. If we use the obvious Fourier sampling algorithm, this problem is solvable with 1 quantum query, with success probability asymptotically equal to

$$\frac{2}{\sqrt{2\pi}}\int_c^\infty e^{-x^2/2}x^2dx.$$

On the other hand, it is a plausible conjecture that any classical randomized algorithm that makes $o\left(N/\log N\right)$ queries to f, can solve the relation problem with probability at most about

$$\frac{2}{\sqrt{2\pi}}\int_c^\infty e^{-x^2/2}dx.$$

If (say) $c=1$, this would give us an 0.8 versus 0.317 gap in success probabilities. That gap could be boosted further using amplification (which, however, would increase the quantum query complexity, from 1 to some larger constant).

8 Appendix: Estimator for Arbitrary Bounded Quadratics

Assume that we have an arbitrary degree-k polynomial

$$p\left(x_1,\ldots,x_N\right)=\sum_{I\subseteq[N]:|I|\leq k} a_I\prod_{i\in I}x_i,$$

with $p\left(x\right)\in[-1,1]$ whenever $x\in\{-1,1\}^N$. We would like to show that $p\left(x\right)$ can be estimated by a randomized algorithm that makes $O\left(N^{1-1/k}\right)$ queries, using a sampling procedure similar to what we used in Section 5 for the special case of block-multilinear polynomials. For our previous proof to work, we need there to exist a sequence of variable-splittings that introduces $O(N)$ new variables, and that transforms $p(x_1,\ldots,x_N)$ into a polynomial

$$q(x_1,\ldots,x_M)=\sum_{I\subseteq[M]:|I|\leq k} b_I\prod_{i\in I}x_i$$

that satisfies the following two requirements:

(i) $\sum_I b_I^2=O(\frac{1}{N})$;

(ii) for all $l\in[k-1]$, we have

$$\sum_{I,J:,I\neq J,|I\cap J|=l} b_Ib_J=O\left(\frac{1}{N^{l/k}}\right). \qquad (10)$$

Requirement (i) is for the bound on the variance of the "warmup estimator," in Section 5.3. Requirement (ii) is for the "real estimator," in Section 5.4. Below, we will be able to prove requirement (i) for any k, and requirement (ii) in the special case $k=2$.

8.1 Fourier Basics

Given a real polynomial $p : \{-1,1\}^N \to \mathbb{R}$, we consider the following notions:

$$\begin{aligned}\mathrm{Var}[p] &:= \mathrm{E}\left[(p(x) - \mathrm{E}[p(x)])^2\right],\\ \mathrm{Inf}_i[p] &:= \mathrm{E}\left[\left(p\left(x^i\right) - p(x)\right)^2\right],\\ \|p\|_\infty &:= \max_{x \in \{-1,1\}^N} |p(x)|\end{aligned}$$

(where x^i means x with the i^{th} bit flipped). Also, let $\hat{p}(S)$ be the Fourier coefficient corresponding to the subset $S \subseteq [N]$—or equivalently, the coefficient in p of the monomial $\prod_{i \in S} x_i$.

Note that, since $\sum_I b_I^2 = \mathrm{Var}[q]$, requirement (i) is equivalent to $\mathrm{Var}[q] = O(\frac{1}{N})$.

From elementary Fourier analysis, we have the following useful lemma.

Lemma 28 *If $p : \{-1,1\}^N \to \mathbb{R}$ is a real polynomial of degree k, then*

$$\sum_{i \in [N]} \mathrm{Inf}_i[p] \leq k \,\mathrm{Var}[p].$$

Proof. We have

$$\mathrm{Inf}_i[p] = \sum_{S \ni i} \hat{p}(S)^2,$$

and hence

$$\sum_{i \in [N]} \mathrm{Inf}_i[p] = \sum_{|S| \leq k} |S|\, \hat{p}(S)^2 \leq k \sum_{|S| \leq k} \hat{p}(S)^2 = k\,\mathrm{Var}[p].$$

■

8.2 Requirement (i)

Our goal is to find variables x_i in p with large influences (that is, large values of $\mathrm{Inf}_i[p]$). To do so, we will use the following result of Dinur et al. [14, Theorem 3].

Theorem 29 ([14], Theorem 3) *There exists a constant C for which the following holds. Suppose $p : \{-1,1\}^N \to \mathbb{R}$ is a real polynomial of degree k, which satisfies $\mathrm{Var}[p] = 1$ and $\mathrm{Inf}_i[p] \leq t^2 C^{-k}$ for all $i \in [N]$. Then*

$$\Pr_{x \in \{-1,1\}^N} [|p(x)| \geq t] \geq \exp(-Ct^2k^2 \log k).$$

Theorem 29 means, in particular, that if $\mathrm{Var}[p] = 1$ and $\mathrm{Inf}_i[p] \leq t^{-2}C^{-k}$ for all $i \in [N]$, then $\|p\|_\infty \geq t$: in other words, there *exists* an $x \in \{-1,1\}^N$ such that $|p(x)| \geq t$. By rescaling, we can turn this into the following.

Corollary 30 *There exists a constant C for which the following holds. Suppose $p : \{-1,1\}^N \to \mathbb{R}$ is a real polynomial of degree k, which satisfies $\|p\|_\infty \leq 1$ and $\mathrm{Var}[p] \geq \frac{C}{N}$. Then there exists an $i \in [N]$ such that $\mathrm{Inf}_i[p] \geq \frac{1}{C^k}\mathrm{Var}[p]^2$.*

Proof. We can reword Theorem 29 as follows: if $\|p\|_\infty \leq t$ and $\mathrm{Var}[p] = 1$, then there exists an $i \in [N]$ such that $\mathrm{Inf}_i[p] \geq \frac{t^2}{C^k}$. Now suppose $\|p\|_\infty \leq 1$. Then define a new polynomial $q := \frac{p}{\sqrt{\mathrm{Var}[p]}}$. We have $\mathrm{Var}[q] = 1$ and $\|q\|_\infty \leq \frac{1}{\sqrt{\mathrm{Var}[p]}}$, which implies that there exists an $i \in [N]$ such that $\mathrm{Inf}_i[q] \geq \frac{\mathrm{Var}[p]}{C^k}$, or equivalently $\mathrm{Inf}_i[p] \geq \frac{\mathrm{Var}[p]^2}{C^k}$. ■

We now consider the following algorithm for variable-splitting. We start with $p_0 := p$. We then repeat the following, for $j \in \{0, 1, 2, \ldots\}$:

(1) If $\mathrm{Var}[p_j] < \frac{C}{N}$, then halt and output p_j.

(2) Otherwise, choose some variable x_i such that $\mathrm{Inf}_i[p] \geq \frac{1}{C^k} \mathrm{Var}[p]^2$ (which is guaranteed to exist by Corollary 30). Let p_{j+1} be the polynomial obtained from p_j by splitting x_i into two variables. (In other words, by defining new variables $x_{i,1}$ and $x_{i,2}$, and then replacing every occurrence of x_i in p_j by $\frac{x_{i,1}+x_{i,2}}{2}$.)

When the algorithm halts (say at step J), we must have $\mathrm{Var}[p_J] < \frac{C}{N}$. Furthermore, observe that for every j,

$$\begin{aligned}\mathrm{Var}[p_{j+1}] &= \mathrm{Var}[p_j] - \frac{\mathrm{Inf}_i[p_j]}{2} \\ &\leq \mathrm{Var}[p_j] - \frac{\mathrm{Var}[p_j]^2}{2C^k}.\end{aligned}$$

Solving this recurrence (together with the initial condition $\mathrm{Var}[p_0] \leq 1$) implies that the algorithm can continue for at most $O(N)$ steps until $\mathrm{Var}[p_J] = O(1/N)$. Therefore, the algorithm introduces at most $O(N)$ new variables.

8.3 Requirement (ii)

We now show how to satisfy requirement (ii) in the special case $k = 2$. To do so, we will need another result from Dinur et al. [14].

Lemma 31 ([14], Lemma 1.3) *There exists a constant C such that the following holds. Let $p : \{-1,1\}^N \to \mathbb{R}$ be a polynomial of degree k, and suppose that*

$$\sum_{i\in[N]} \hat{p}(\{i\})^2 \geq 1$$

whereas $|\hat{p}(\{i\})| \leq \frac{1}{Ckt}$ for all $i \in [N]$. Then

$$\Pr_{x\in\{-1,1\}^N}[|p(x)| \geq t] \geq \exp(-Ct^2k^2).$$

By rescaling p (similarly to Corollary 30), we can transform Lemma 31 into the following.

Corollary 32 *There exists a constant C such that the following holds. For any bounded real polynomial $p : \{-1,1\}^N \to [-1,1]$ of degree k that satisfies*

$$\sum_{i\in[N]} \hat{p}(\{i\})^2 = v,$$

there exists an $i \in [N]$ such that $|\hat{p}(\{i\})| \geq \frac{v}{Ck}$.

Proof. Let us define a new polynomial $q(x) := \frac{p(x)}{\sqrt{v}}$. Then $q(x) \in \left[-\frac{1}{\sqrt{v}}, \frac{1}{\sqrt{v}}\right]$ for all $x \in \{-1,1\}^N$, and

$$\sum_{i\in[N]} \hat{q}(\{i\})^2 = 1.$$

So by Lemma 31, there must exist an index $i \in [N]$ such that

$$|\hat{q}(\{i\})| > \frac{\sqrt{v}}{Ck}$$

and hence

$$|\hat{p}(\{i\})| > \frac{v}{Ck}.$$

■

We now specialize to the case $k = 2$. Observe that, given a bounded quadratic polynomial $p : \{-1,1\}^N \to [-1,1]$ that we are trying to estimate, we can assume without loss of generality that p is homogeneous: in other words, is a quadratic form. First of all, if p contains a degree-0 term (i.e., an additive constant) c, then we can replace it by the degree-2 term cx_1^2. Next, suppose we decompose p as a sum of its quadratic part p_2 and its linear part p_1:

$$p(x) = p_2(x) + p_1(x).$$

Then

$$p(-x) = p_2(x) - p_1(x)$$

is also bounded in $[-1,1]$ for all $x \in \{-1,1\}^N$. So

$$p_1(x) = \frac{p(x) - p(-x)}{2} \in [-1,1],$$

and likewise $p_2(x) \in [-1,1]$, for all $x \in \{-1,1\}^N$. Hence we can simply estimate $p_2(x)$ and $p_1(x)$ separately and then sum them. Furthermore, p_1 has the form

$$p_1(x) = a_1x_1 + \cdots + a_nx_n$$

for some coefficients satisfying $|a_1| + \cdots + |a_n| \leq 1$. By standard tail bounds, such a linear form is easy to estimate to within $\pm\varepsilon$ by querying only $O(1)$ variables x_i.

So our problem reduces to estimating the bounded quadratic form $p_2(x)$. To do this, the first step is to apply the variable-splitting algorithm described in Section 8.2. This produces a *new* bounded quadratic form, which we denote P:

$$P(x_1,\ldots,x_N) = \sum_{i,j\in[N]} a_{i,j}x_ix_j.$$

(Abusing notation, we continue to call the number of variables N, even though $O(N)$ new variables have been introduced.) We have $\mathrm{Var}[P] = O(\frac{1}{N})$. Now, our goal is to perform another sequence of variable-splittings, which should introduce $O(N)$ new variables and achieve requirement (ii).

Observe that, to achieve requirement (ii), it suffices to ensure

$$\sum_{i\in[N]} \left(\sum_{i\neq j} a_{ij}\right)^2 = O\left(\frac{1}{\sqrt{N}}\right). \tag{11}$$

For if we achieve (11), then we can obtain the original requirement (ii) by removing all squares a_{ij}^2 from the left hand side (which only decreases it) and dividing it by 2.

Let

$$\mathrm{I}_i\left[P\right] := \left(\sum_{i\neq j} a_{ij}\right)^2,$$
$$\mathrm{V}\left[P\right] := \sum_{i\in[N]} \mathrm{I}_i\left[P\right].$$

We show the following counterpart of Corollary 30.

Lemma 33 *There exists a constant C for which the following holds. If $\mathrm{V}\left[P\right] \geq \frac{C\log N}{N}$, then there exists an $i\in[N]$ such that $\mathrm{I}_i\left[P\right] = \Omega(\mathrm{V}\left[P\right]^2)$.*

Proof. We take $P(x_1,\ldots,x_N)$ and, for each $i\in[N]$, substitute $x_i = 1$ with probability 1/2 (with choices for different i made independently). Let q be the resulting polynomial.

The key claim is the following: let x_i be a variable for which we do not substitute $x_i = 1$. Then

$$\left|\hat{q}\left(\{i\}\right) - \frac{1}{2}\sum_{i\neq j} a_{ij}\right| = O\left(\sqrt{\mathrm{Inf}_i[P]\log\frac{1}{\epsilon}}\right) \tag{12}$$

with probability at least $1-\epsilon$.

To prove the claim: since each term x_ix_j is transformed into x_i with probability 1/2 by substituting $x_i = 1$, the expectation of $\hat{q}\left(\{i\}\right)$ is equal to $\frac{1}{2}\sum_{j\neq i} a_{ij}$. Since the decision to substitute or not substitute $x_i = 1$ changes the value of $\hat{q}\left(\{i\}\right)$ by the amount a_{ij}, Azuma's inequality implies

$$\Pr\left[\left|\hat{q}\left(\{i\}\right) - \frac{1}{2}\sum_{i\neq j} a_{ij}\right| \geq t\right] \leq \exp\left(-\frac{t^2}{2\sum_{j\in[N]} a_{ij}^2}\right).$$

Using $\mathrm{Inf}_i[P] = \sum_{j\in[N]} a_{ij}^2$ and taking $t = \sqrt{2\,\mathrm{Inf}_i[P]\log\frac{1}{\epsilon}}$ completes the proof.

We now show how the claim implies the lemma. Let $\epsilon = \frac{1}{2N}$. Then we can make the substitutions so that (12) holds for all $i\in[N]$. By substituting $\epsilon = \frac{1}{2N}$ into (12), we have

$$\left|\hat{q}\left(\{i\}\right)\right| \in \left[\frac{\sqrt{\mathrm{I}_i\left[P\right]}}{2} - O\left(\sqrt{\mathrm{Inf}_i\left[P\right]\log N}\right), \frac{\sqrt{\mathrm{I}_i\left[P\right]}}{2} + O\left(\sqrt{\mathrm{Inf}_i\left[P\right]\log N}\right)\right] \tag{13}$$

By squaring (13) and summing over all $i\in[N]$, we obtain

$$\begin{aligned}
\left|\sum_{i\in[N]} \hat{q}\left(\{i\}\right)^2 - \frac{1}{4}\sum_{i\in[N]} \mathrm{I}_i\left[P\right]\right| &= O\left(\sqrt{\log N}\right)\sum_{i\in[N]} \sqrt{\mathrm{I}_i\left[P\right]\mathrm{Inf}_i\left[P\right]} + O\left(\log N\right)\sum_{i\in[N]} \mathrm{Inf}_i\left[P\right] \\
&\leq O\left(\sqrt{\log N}\right)\sqrt{\sum_{i\in[N]} \mathrm{I}_i\left[P\right]\cdot\sum_{i\in[N]} \mathrm{Inf}_i\left[P\right]} + O\left(\log N\right)\sum_{i\in[N]} \mathrm{Inf}_i\left[P\right] \\
&\leq O\left(\sqrt{\log N}\right)\sqrt{\mathrm{V}\left[P\right]\cdot 2\,\mathrm{Var}\left[P\right]} + O\left(\log N\right)\cdot 2\,\mathrm{Var}\left[P\right] \\
&= O\left(\sqrt{\mathrm{V}\left[P\right]\frac{\log N}{N}} + \frac{\log N}{N}\right),
\end{aligned}$$

where the second line used Cauchy-Schwarz and the third used Lemma 28.

Now, if $\mathrm{V}[P] \geq C\frac{\log N}{N}$ for a sufficiently large constant C, then the above implies that

$$\begin{aligned}\sum_{i\in[N]} \hat{q}(\{i\})^2 &\geq \frac{1}{4}\mathrm{V}[P] - O\left(\sqrt{\mathrm{V}[P]\frac{\log N}{N}} + \frac{\log N}{N}\right) \\ &= \Omega(\mathrm{V}[P]).\end{aligned}$$

By Corollary 32, this means that there exists an index $i \in [N]$ such that $|\hat{q}(\{i\})| = \Omega(\mathrm{V}[P])$. From equation (13), we get that $\sqrt{\mathrm{I}_i[P]} = \Omega(\mathrm{V}[P])$ and $\mathrm{I}_i[P] = \Omega(\mathrm{V}[P]^2)$. ■

Given Lemma 33, we can use the same algorithm as in the previous section. That is, we repeatedly choose a variable $i \in [N]$ that maximizes $\mathrm{I}_i[P]$ and split the variable x_i. Let $p_0, p_1, \ldots$ be the resulting sequence of polynomials. Initially, we have

$$\mathrm{V}[p_0] = \sum_i \left(\sum_{j\neq i} a_{ij}\right)^2 \leq N\sum_i\sum_j a_{ij}^2 = O(1),$$

with the last equality following from $\mathrm{Var}[P] = O(\frac{1}{N})$, which was achieved by the previous sequence of variable-splittings. We also have

$$\mathrm{V}[p_{j+1}] \leq \mathrm{V}[p_j] - \Omega(\mathrm{V}[p_j]^2),$$

as long as $\mathrm{V}[p_j] \geq \frac{C\log N}{N}$. This means that after $O(N/\log N)$ variable-splittings, we achieve $\mathrm{V}[p_j] < \frac{C\log N}{N}$, which is substantially stronger than the requirement (11) that we needed.

9 Appendix: Lower Bound for k-fold Forrelation

In this appendix, we use the machinery developed in Section 4 to prove a $\widetilde{\Omega}(\sqrt{N})$ lower bound on the randomized query complexity of k-fold FORRELATION, for all $k \geq 2$. Ideally, of course, we would like to prove a lower bound that gets *better* as k gets larger, but even proving the same kind of lower bound that we had in the $k = 2$ case will take some work.

In k-fold FORRELATION, recall that we are given oracle access to Boolean functions $f_1, \ldots, f_k : \{0,1\}^n \to \{-1,1\}$, and are interested in the quantity

$$\Phi_{f_1,\ldots,f_k} := \frac{1}{2^{(k+1)n/2}} \sum_{x_1,\ldots,x_k\in\{0,1\}^n} f_1(x_1)(-1)^{x_1\cdot x_2} f_2(x_2)(-1)^{x_2\cdot x_3}\cdots(-1)^{x_{k-1}\cdot x_k} f_k(x_k).$$

The problem is to decide whether $|\Phi_{f_1,\ldots,f_k}| \leq \frac{1}{100}$ or $\Phi_{f_1,\ldots,f_k} \geq \frac{3}{5}$, promised that one of these is the case.

We will prove a lower bound for k-fold FORRELATION by reducing the GAUSSIAN DISTINGUISHING problem to it, and then applying Theorem 11—thereby illustrating the usefulness of formulating a general lower bound for GAUSSIAN DISTINGUISHING.

9.1 Concentration Inequalities

The first step is to prove some concentration inequalities for k-fold Forrelation. In what follows, recall that $f_\ell^{(x)}(x_\ell)$ is defined as $f_\ell(x_\ell)(-1)^{x_\ell \cdot x}$, for all $\ell \in [k]$ and $x_\ell, x \in \{0,1\}^n$.

Lemma 34 *Suppose $f_1, \ldots, f_k : \{0,1\}^n \to \{-1,1\}$ are chosen uniformly at random. Then*

$$\Pr_{f_1,\ldots,f_k}\left[\left|\Phi_{f_1,\ldots,f_k}\right| \geq \frac{t}{\sqrt{N}}\right] = O\left(\frac{1}{t^t}\right).$$

Proof. Imagine that $f_1, \ldots, f_{k-1}$ are fixed, so that we are considering $\Phi_{f_1,\ldots,f_k}$ solely as a function of f_k. We have

$$\Phi_{f_1,\ldots,f_k} = \frac{1}{\sqrt{N}} \sum_{x \in \{0,1\}^n} \alpha_x f_k(x)$$

where

$$\alpha_x = \Phi_{f_1,\ldots,f_{k-2},f_{k-1}^{(x)}} = \frac{1}{2^{kn/2}} \sum_{x_1,\ldots,x_{k-1} \in \{0,1\}^n} f_1(x_1)(-1)^{x_1 \cdot x_2} \cdots (-1)^{x_{k-2} \cdot x_{k-1}} f_{k-1}(x_{k-1})(-1)^{x_{k-1} \cdot x}.$$

Thus, by equation (1),

$$\sum_{x \in \{0,1\}^n} \alpha_x^2 = \sum_{x \in \{0,1\}^n} \Phi^2_{f_1,\ldots,f_{k-2},f_{k-1}^{(x)}} = 1.$$

So in particular, $|\alpha_x| \leq 1$ for all x. We now appeal to Bennett's inequality, which tells us that

$$\Pr_{f_k:\{0,1\}^n \to \{-1,1\}}\left[\left|\sum_{x \in \{0,1\}^n} \alpha_x f_k(x)\right| \geq t\right] \leq 2\exp(-h(t))$$

where

$$h(t) := (1+t)\ln(1+t) - t.$$

■

The following is also useful.

Lemma 35 *Suppose $f_1, \ldots, f_k : \{0,1\}^n \to \{-1,1\}$ are chosen uniformly at random. Then with probability $1 - O(1/N)$, we have*

$$\left|\sum_{z\ :\ z \cdot y = 0} \Phi^2_{f_1,\ldots,f_{k-1},f_k^{(z)}} - \frac{1}{2}\right| \leq \frac{\log^{5/2} N}{\sqrt{N}}$$

for all $y \in \{0,1\}^n$.

Proof. By symmetry, we can assume without loss of generality that $y = 10\cdots 0$, so that the sum is over all z that start with 0.

As in the proof of Lemma 34, imagine that $f_1, \ldots, f_{k-1}$ are fixed, so that we are considering

$$\Phi_{f_1,\ldots,f_{k-1},f_k^{(z)}} = \frac{1}{\sqrt{N}} \sum_{x \in \{0,1\}^n} \alpha_x f_k(x)(-1)^{x \cdot z}$$

solely as a function of f_k. Then it is not hard to see that

$$\sum_{z\ :\ z\cdot y=0} \Phi^2_{f_1,\ldots,f_{k-1},f_k^{(z)}}$$

is just the probability of measuring an output string that starts with 0 when the Forrelation algorithm is run on $f_1,\ldots,f_k$, which also equals

$$\begin{aligned}\sum_{x\in\{0,1\}^{n-1}} \left(\frac{\alpha_{0x} f_k(0x) + \alpha_{1x} f_k(1x)}{\sqrt{2}}\right)^2 &= \frac{1}{2}\sum_{x\in\{0,1\}^{n-1}} \left(\alpha_{0x} f_k(0x) + \alpha_{1x} f_k(1x)\right)^2 \\ &= \frac{1}{2}\sum_{x\in\{0,1\}^{n-1}} \left(\alpha_{0x}^2 + \alpha_{1x}^2 + 2\alpha_{0x}\alpha_{1x} f_k(0x) f_k(1x)\right) \\ &= \frac{1}{2} + \sum_{x\in\{0,1\}^{n-1}} \alpha_{0x}\alpha_{1x} f_k(0x) f_k(1x).\end{aligned}$$

Now, each $f_k(0x) f_k(1x)$ is an independent, uniform $\{-1,1\}$ random variable. Furthermore, by Lemma 34, with $1 - o(1/N)$ probability we have $|\alpha_x| \le \frac{\log N}{\sqrt{N}}$ for all $x \in \{0,1\}^n$, in which case

$$|\alpha_{0x}\alpha_{1x}| \le \frac{\log^2 N}{N}$$

for all $x \in \{0,1\}^{n-1}$. By Hoeffding's inequality, it follows that

$$\begin{aligned}\Pr\left[\left|\sum_{z\ :\ z\cdot y=0} \Phi^2_{f_1,\ldots,f_{k-1},f_k^{(z)}} - \frac{1}{2}\right| \ge \frac{t}{\sqrt{N}}\right] &\le 2\exp\left(-\frac{2(t/\sqrt{N})^2}{(N/2)\left(\frac{2\log^2 N}{N}\right)^2}\right) \\ &= 2\exp\left(-\frac{t^2}{4\log^4 N}\right).\end{aligned}$$

Setting $t = C\log^{5/2} N$ for some constant C, this probability is at most $1/N^2$. So by the union bound, the probability is at most $1/N$ when summed over all y. ■

9.2 Continuous/Discrete Hybrid

Just like we did in the $k = 2$ case, it is convenient to define a continuous analogue of the k-fold Forrelation problem—though in this case, the problem will be a hybrid of continuous and discrete. In k-fold Real Forrelation, we are given oracle access to functions $f_1,\ldots,f_{k-2} : \{0,1\}^n \to \{-1,1\}$ as well as $f_{k-1}, f_k : \{0,1\}^n \to \mathbb{R}$. We are promised that each $f_i(x_i)$ (for $i \in [k-2]$) is chosen uniformly and independently from $\{-1,1\}$, and *also* that one of the following holds:

(i) Uniform measure $\mathcal{U}$: Each $f_{k-1}(x_{k-1})$ and $f_k(x_k)$ is an independent $\mathcal{N}(0,1)$ Gaussian.

(ii) Forrelated measure $\mathcal{F}$: Each $f_{k-1}(x_{k-1})$ is an independent $\mathcal{N}(0,1)$ Gaussian, and each $f_k(x_k)$ is set equal to

$$\frac{1}{2^{(k-1)n/2}} \sum_{x_1,\ldots,x_{k-1}\in\{0,1\}^n} f_1(x_1)(-1)^{x_1\cdot x_2}\cdots(-1)^{x_{k-2}\cdot x_{k-1}} f_{k-1}(x_{k-1})(-1)^{x_{k-1}\cdot x_k}.$$

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

Here is another way to think about k-fold Real Forrelation: let

$$f\left(x_{k-1}\right) := c_{x_{k-1}} f_{k-1}\left(x_{k-1}\right), \qquad g\left(x_k\right) := f_k\left(x_k\right),$$

where

$$\begin{aligned} c_{x_{k-1}} &= \sqrt{N}\Phi_{f_1,\ldots,f_{k-3},f_{k-2}^{\left(x_{k-1}\right)}} \\ &= \frac{1}{2^{(k-2)n/2}} \sum_{x_1,\ldots,x_{k-2}\in\{0,1\}^n} f_1\left(x_1\right)(-1)^{x_1\cdot x_2}\cdots f_1\left(x_{k-2}\right)(-1)^{x_{k-2}\cdot x_{k-1}}. \end{aligned}$$

Then

$$\Phi_{f_1,\ldots,f_k} = \Phi_{f,g} = \frac{1}{2^{3n/2}} \sum_{x,y\in\{0,1\}^n} f\left(x\right)(-1)^{x\cdot y} g\left(y\right).$$

In other words, we can think of k-fold Real Forrelation as equivalent to ordinary, 2-fold Real Forrelation on the functions $f_{k-1}\left(x\right)$ and $f_k\left(y\right)$, except that each $f_{k-1}\left(x\right)$ is "twisted" by a multiplicative factor c_x depending on $f_1,\ldots,f_{k-2}$. We have the following useful fact.

Proposition 36 *With $1-o\left(1/N\right)$ probability over $f_1,\ldots,f_{k-2}$, we have $\left|c_x\right| \leq \log N$ for all $x \in \{0,1\}^n$.*

Proof. By Lemma 34,

$$\Pr_{f_1,\ldots,f_{k-2}}\left[\left|c_x\right| \geq \log N\right] = O\left(\frac{1}{\left(\log N\right)^{\log N}}\right) = o\left(\frac{1}{N^2}\right).$$

The proposition now follows from the union bound. ■

Note also that

$$\sum_{x\in\{0,1\}^n} c_x^2 = N.$$

Next, we prove a k-fold analogue of Theorem 9, showing that k-fold Real Forrelation can be reduced to Boolean k-fold Forrelation.

Theorem 37 *Fix $f_1,\ldots,f_{k-2}$ (or equivalently, the multipliers c_x). Suppose $\langle f,g\rangle = \langle f_{k-1},f_k\rangle$ are drawn from the forrelated measure $\mathcal{F}$. Define Boolean functions $F,G:\{0,1\}^n \rightarrow \{-1,1\}$ by $F\left(x\right) := \operatorname{sgn}\left(f\left(x\right)\right)$ and $G\left(y\right) := \operatorname{sgn}\left(g\left(y\right)\right)$. Then*

$$\operatorname{E}\left[\Phi_{F,G}\right] = \frac{2}{\pi} \pm O\left(\frac{\log^3 N}{N}\right).$$

Proof. By Proposition 36, we can assume without loss of generality that $\left|c_x\right| \leq \log N$ for all $x \in \{0,1\}^n$ (the times when this assumption fails can only change $\operatorname{E}\left[\Phi_{F,G}\right]$ by $o\left(1/N\right)$).

By linearity of expectation, it suffices to calculate $\mathrm{E}\left[c_x F(x)(-1)^{x\cdot y} G(y)\right]$ for some specific x, y pair. Let $v \in \mathbb{R}^N$ be a vector of independent $\mathcal{N}(0,1)$ Gaussians. Then we can consider $\langle F, G\rangle$ to have been generated as follows:

$$\begin{aligned} F(x) &= \operatorname{sgn}(v_x), \\ G(y) &= \operatorname{sgn}\left(\sum_{x\in\{0,1\}^n} c_x v_x (-1)^{x\cdot y}\right). \end{aligned}$$

Let

$$Z := \sum_{x'\neq x} c_{x'} v_{x'} (-1)^{x'\cdot y},$$

and let $G'(y) := \operatorname{sgn}(Z)$. Then

$$\mathrm{E}\left[c_x F(x)(-1)^{x\cdot y} G'(y)\right] = \mathrm{E}\left[c_x \operatorname{sgn}(v_x)(-1)^{x\cdot y} \operatorname{sgn}(Z)\right] = 0,$$

since v_x and Z are independent Gaussians both with mean 0. Note that adding $c_x v_x (-1)^{x\cdot y}$ back to Z can only flip Z to having the same sign as $c_x \operatorname{sgn}(v_x)(-1)^{x\cdot y}$, not the opposite sign, and hence can only increase $c_x F(x)(-1)^{x\cdot y} G(y)$. It follows that

$$\mathrm{E}\left[c_x F(x)(-1)^{x\cdot y} G(y)\right] = 2\Pr\left[G(y) \neq G'(y)\right].$$

The event $G(y) \neq G'(y)$ occurs if and only if the following two events both occur:

$$\begin{gathered} |c_x v_x| > |Z|, \\ \operatorname{sgn}\left(c_x v_x (-1)^{x\cdot y}\right) \neq \operatorname{sgn}(Z). \end{gathered}$$

Since the distribution of v_x is symmetric about 0, we can assume without loss of generality that $c_x (-1)^{x\cdot y} = 1$.

Let $Z(t)$ be the probability density function of Z. Then

$$\Pr\left[|c_x v_x| > |Z| \quad \text{and} \quad \operatorname{sgn}(v_x) \neq \operatorname{sgn}(Z)\right] = 2\int_{t=0}^{\infty} Z(t) \Pr\left[c_x v_x > t\right] dt.$$

Now, Z is a linear combination of $N-1$ independent $\mathcal{N}(0,1)$ Gaussians, with coefficients $\{c_{x'}\}_{x'\neq x}$. This means that Z has the $\mathcal{N}\left(0, N - c_x^2\right)$ Gaussian distribution. Therefore

$$\begin{aligned} 2\int_{t=0}^{\infty} Z(t)\Pr\left[c_x v_x > t\right] dt &= \frac{2}{\sqrt{2\pi\left(N - c_x^2\right)}} \int_{t=0}^{\infty} \exp\left(-\frac{t^2}{2\left(N - c_x^2\right)}\right) \Pr\left[c_x v_x > t\right] dt \\ &\leq \frac{2}{\sqrt{2\pi\left(N - c_x^2\right)}} \int_{t=0}^{\infty} \Pr\left[c_x v_x > t\right] dt \\ &= \frac{2}{\sqrt{2\pi\left(N - c_x^2\right)}} \mathrm{E}\left[|c_x v_x|\right] \\ &= \frac{2|c_x|}{\pi\sqrt{N - c_x^2}} \\ &\leq \frac{2|c_x|}{\pi\sqrt{N}} + O\left(\frac{|c_x|^3}{N^{3/2}}\right). \end{aligned}$$

In the other direction, for all $C > 0$ we have

$$\begin{aligned}
2\int_{t=0}^{\infty} Z(t)\Pr[c_x v_x > t]\,dt &= \frac{2}{\sqrt{2\pi(N-c_x^2)}}\int_{t=0}^{\infty}\exp\left(-\frac{t^2}{2(N-c_x^2)}\right)\Pr[c_x v_x > t]\,dt \\
&\geq \frac{2}{\sqrt{2\pi N}}\int_{t=0}^{C}\exp\left(-\frac{t^2}{2(N-c_x^2)}\right)\Pr[c_x v_x > t]\,dt \\
&\geq \frac{2}{\sqrt{2\pi N}}\exp\left(-\frac{C^2}{2(N-c_x^2)}\right)\int_{t=0}^{C}\Pr[c_x v_x > t]\,dt \\
&= \frac{2}{\sqrt{2\pi N}}\exp\left(-\frac{C^2}{2(N-c_x^2)}\right)\left(\mathrm{E}[|c_x v_x|] - \frac{1}{c_x\sqrt{2\pi}}\int_{t=C}^{\infty} te^{-t^2/(2c_x^2)}dt\right) \\
&= \frac{2}{\sqrt{2\pi N}}\exp\left(-\frac{C^2}{2(N-c_x^2)}\right)|c_x|\left(\sqrt{\frac{2}{\pi}} - \frac{e^{-C^2/2}}{\sqrt{2\pi}}\right).
\end{aligned}$$

If we set $C := \sqrt{\log N}$, then using $|c_x| \leq \log N$, the above is

$$\frac{2|c_x|}{\sqrt{2\pi N}}\left(1 - O\left(\frac{\log N}{N}\right)\right)\left(\sqrt{\frac{2}{\pi}} - \frac{1}{\sqrt{2\pi N}}\right) \geq \frac{2|c_x|}{\pi\sqrt{N}} - O\left(\frac{\log N}{N^{3/2}}\right).$$

Therefore

$$\begin{aligned}
\mathrm{E}[\Phi_{F,G}] &= \frac{1}{2^{3n/2}}\sum_{x,y\in\{0,1\}^n}\mathrm{E}\left[c_x F(x)(-1)^{x\cdot y}G(y)\right] \\
&= \frac{1}{N^{3/2}}\sum_{x,y\in\{0,1\}^n}\left(\frac{2|c_x|}{\pi\sqrt{N}} + O\left(\frac{|c_x|^3}{N^{3/2}}\right) - O\left(\frac{\log N}{N^{3/2}}\right)\right) \\
&= \frac{2}{\pi} + \frac{1}{N^2}\sum_{x\in\{0,1\}^n}|c_x|^3 - O\left(\frac{\log N}{N}\right)\backslash \\
&= \frac{2}{\pi} \pm O\left(\frac{\log^3 N}{N}\right).
\end{aligned}$$

■

By direct analogy to Corollary 10, Theorem 37 implies that there exists a reduction from k-fold Real Forrelation to k-fold Forrelation.

Corollary 38 *Suppose there exists a T-query algorithm that solves k-fold* Forrelation *with bounded error. Then there also exists an $O(T)$-query algorithm that solves k-fold* Real Forrelation *with bounded error.*

9.3 Lower Bound

Finally, we apply our lower bound for Gaussian Distinguishing to obtain a lower bound on the randomized query complexity of k-fold Real Forrelation, which almost matches what we obtained for the 2-fold case.

Theorem 39 *k-fold* Real Forrelation *requires $\Omega(\sqrt{N}/\log^{7/2} N)$ randomized queries.*

Proof. The strategy is the following: we will give the values $f_1(x_1), \ldots, f_{k-2}(x_{k-2})$, for all $x_1, \ldots, x_{k-2} \in \{0,1\}^n$, away to the algorithm "free of charge." This can only make our lower bound stronger.

As we saw above, after we do this, k-fold Real Forrelation becomes equivalent to ordinary 2-fold Real Forrelation on the functions $f(x) = c_x f_{k-1}(x)$ and $g(y) = f_k(y)$, where the c_x's are known multipliers. This, in turn, can be expressed as an instance of Gaussian Distinguishing, in which our set $\mathcal{V}$ of test vectors consists of $|1\rangle, \ldots, |N\rangle$ along with the vectors $\{|\psi_y\rangle\}_{y\in\{0,1\}^n}$ defined as follows:

$$|\psi_y\rangle := \frac{1}{\sqrt{N}} \sum_{x\in\{0,1\}^n} c_x (-1)^{x\cdot y} |x\rangle .$$

Note that the $|\psi_y\rangle$'s are unit vectors, since $\sum_x c_x^2 = N$. As for inner products, with $1 - O(1/N)$ probability we have

$$|\langle x|\psi_y\rangle| = \frac{|c_x|}{\sqrt{N}} \leq \frac{\log N}{\sqrt{N}}$$

for all x, y by Proposition 36, and

$$|\langle\psi_y|\psi_z\rangle| = \frac{1}{N}\left|\sum_{x\in\{0,1\}^n} c_x^2 (-1)^{x\cdot(y\oplus z)}\right| = O\left(\frac{\log^{5/2} N}{\sqrt{N}}\right)$$

for all y, z by Lemma 35. So, in summary, we have $|\mathcal{V}| = 2N$ and $|\langle v|w\rangle| \leq \varepsilon = O\left(\frac{\log^{5/2} N}{\sqrt{N}}\right)$ for all distinct $|v\rangle, |w\rangle \in \mathcal{V}$. By Theorem 11, this implies that

$$\Omega\left(\frac{1/\varepsilon}{\log(2N/\varepsilon)}\right) = \Omega\left(\frac{\sqrt{N}}{\log^{7/2} N}\right)$$

queries are needed. ■

10 Appendix: Property Testing

In this appendix, we show that Forrelation can be recast as a *property-testing* problem: that is, as a problem of deciding whether the functions $f, g : \{0,1\}^n \to \{-1,1\}$ satisfy a certain property, or are far in Hamming distance from any functions satisfying that property. (For a recent survey of quantum property-testing, see Montanaro and de Wolf [17].)

In particular, we will obtain a property of N-bit strings that

(1) can be quantumly ε-tested (with bounded error) using only $O(1/\varepsilon)$ queries, but

(2) requires $\Omega(\frac{\sqrt{N}}{\log N})$ queries to ε-test classically, provided ε is a sufficiently small constant.

This is the largest quantum versus classical property-testing separation yet known.

Since our analysis works for k-fold Forrelation just as easily as for 2-fold, we will use the more general setting. Let $\mathcal{Y}$ be the set of all k-tuples of Boolean functions $\langle f_1, \ldots, f_k\rangle$ such that $\Phi_{f_1,\ldots,f_k} \leq \frac{1}{100}$. Also, let $\mathcal{N}_\varepsilon$ be the set of all k-tuples $\langle g_1, \ldots, g_k\rangle$ that differ from every $\langle f_1, \ldots, f_k\rangle \in \mathcal{Y}$ on at least $\varepsilon \cdot k2^n$ of the $k2^n$ positions. Then we will be interested in the problem of deciding whether $\langle f_1, \ldots, f_k\rangle \in \mathcal{Y}$ or $\langle f_1, \ldots, f_k\rangle \in \mathcal{N}_\varepsilon$, promised that one of these is the case.

Our goal is to show that both our quantum algorithm for FORRELATION, *and* our randomized lower bound for it, carry over to the property-testing variant. There are two difficulties here. First, one could imagine an $\langle f_1, \ldots, f_k \rangle$ that was far from $\mathcal{Y}$ in the Hamming distance sense, yet had an $\Phi_{f_1,\ldots,f_k}$ value only *slightly* greater than $\frac{1}{100}$—in which case, we would need many repetitions of our quantum algorithm to separate $\langle f_1, \ldots, f_k \rangle$ from $\mathcal{Y}$. Second, one could imagine an $\langle f_1, \ldots, f_k \rangle$ that was close to $\mathcal{Y}$ in Hamming distance, yet had (say) $\Phi_{f_1,\ldots,f_k} \geq \frac{3}{5}$—in which case, the known classical hardness of distinguishing $|\Phi_{f_1,\ldots,f_k}| \leq \frac{1}{100}$ from $\Phi_{f_1,\ldots,f_k} \geq \frac{3}{5}$ might not imply anything about the hardness of distinguishing $\langle f_1, \ldots, f_k \rangle \in \mathcal{Y}$ from $\langle f_1, \ldots, f_k \rangle \in \mathcal{N}_\varepsilon$, causing the classical lower bound to fail.[18]

Fortunately, we can deal with both difficulties. To start with the first:

Lemma 40 *Let $f_1, \ldots, f_k : \{0,1\}^n \to \{-1,1\}$ be Boolean functions satisfying $\Phi_{f_1,\ldots,f_k} \geq 0$. Then for all $\varepsilon > 0$, there exist functions $g_1, \ldots, g_k$ such that each g_i differs from f_i on at most $\varepsilon 2^n$ coordinates, and $\Phi_{g_1,\ldots,g_k} \leq (1-\varepsilon)^k \Phi_{f_1,\ldots,f_k}$.*

Proof. We form each g_i by simply choosing a subset $S_i \subset \{0,1\}^n$ with $|S_i| = \varepsilon 2^n$ uniformly at random, then picking $g_i(x)$ uniformly at random if $x \in S_i$, or setting $g_i(x) := f_i(x)$ if $x \notin S_i$. By linearity of expectation,

$$\mathrm{E}\left[\Phi_{g_1,\ldots,g_k}\right] = \frac{1}{2^{(k+1)n/2}} \sum_{x_1,\ldots,x_k \in \{0,1\}^n} \mathrm{E}\left[g_1(x_1)(-1)^{x_1 \cdot x_2} g_2(x_2)(-1)^{x_2 \cdot x_3} \cdots (-1)^{x_{k-1} \cdot x_k} g_k(x_k)\right].$$

By symmetry, the expectation inside the sum is 0 if we condition on $x_i \in S_i$ for any i. Conversely, if we condition on $x_i \notin S_i$ for all i, then the expectation is

$$\Lambda_{x_1,\ldots,x_k} := f_1(x_1)(-1)^{x_1 \cdot x_2} f_2(x_2)(-1)^{x_2 \cdot x_3} \cdots (-1)^{x_{k-1} \cdot x_k} f_k(x_k).$$

Overall, then, the expectation is

$$\Lambda_{x_1,\ldots,x_k} \cdot \prod_{i \in [k]} \Pr_{S_i}\left[x_i \notin S_i\right] = \Lambda_{x_1,\ldots,x_k} \cdot (1-\varepsilon)^k,$$

which yields

$$\mathrm{E}\left[\Phi_{g_1,\ldots,g_k}\right] = (1-\varepsilon)^k \Phi_{f_1,\ldots,f_k}.$$

Clearly, then, there exists at least one choice of $g_1, \ldots, g_k$ such that

$$\Phi_{g_1,\ldots,g_k} \leq (1-\varepsilon)^k \Phi_{f_1,\ldots,f_k}.$$

■

In contrapositive form, Lemma 40 implies that, if a k-tuple of functions $\langle f_1, \ldots, f_k \rangle$ has Hamming distance at least $\varepsilon \cdot k 2^n$ from every tuple $\langle g_1, \ldots, g_k \rangle$ such that $\Phi_{g_1,\ldots,g_k} \leq c$ (for some $c \geq 0$), then we must have

$$\Phi_{f_1,\ldots,f_k} > \frac{c}{(1-\varepsilon)^k} > c(1+k\varepsilon).$$

[18] Furthermore, this worry is not farfetched: if $k \geq 3$, then there really *are* cases where changing a single function value can change $\Phi_{f_1,\ldots,f_k}$ dramatically. For example, let f_1, f_2, and f_3 each be the identically-1 function. Then $\Phi_{f_1,f_2,f_3} = 1$. But if we simply change $f_2(0^n)$ from 1 to -1, then $\Phi_{f_1,f_2,f_3} = -1$.

As a side note, we conjecture that Lemma 40 also holds "in the other direction"—that is, that if $\langle f_1,\ldots,f_k\rangle$ has Hamming distance at least $\varepsilon\cdot k2^n$ from every $\langle g_1,\ldots,g_k\rangle$ such that $\Phi_{g_1,\ldots,g_k}\geq c$ (for some $c>0$), then $\Phi_{f_1,\ldots,f_k}$ must be significantly *smaller* than c—but we do not currently have a proof of that.

We now show how to deal with the second difficulty. Call a k-tuple of Boolean functions $f_1,\ldots,f_k:\{0,1\}^n\rightarrow\{-1,1\}$ *good* if

$$\left|\Phi_{f_1,\ldots,f_{i-1},f_i^{(x)}}\right|\leq\frac{\log N}{\sqrt{N}}$$

for every $i\in[k]$ and $x\in\{0,1\}^n$; and moreover, there exists a constant C_k such that, for every $i\in[k]$ and $t\in[\log N]$, the "partial sums" $\Phi_{f_1,\ldots,f_{i-1},f_i^{(x)}}$ satisfy the following property:

$$\Pr_{x\in\{0,1\}^n}\left[\left|\Phi_{f_1,\ldots,f_{i-1},f_i^{(x)}}\right|\geq\frac{t}{\sqrt{N}}\right]\leq\frac{C_k}{t^{t/2}}.$$

Then we have the following extension of Proposition 36:

Proposition 41 *If $\langle f_1,\ldots,f_k\rangle$ is chosen uniformly at random, then it is good with probability at least $1-\delta_k$, where δ_k can be made arbitrarily small by increasing C_k.*

Proof. By Lemma 34, for all $i\in[k]$ and $x\in\{0,1\}^n$ we have

$$\Pr_{f_1,\ldots,f_i}\left[\left|\Phi_{f_1,\ldots,f_{i-1},f_i^{(x)}}\right|\geq\frac{t}{\sqrt{N}}\right]=O\left(\frac{1}{t^t}\right).$$

So by Markov's inequality,

$$\Pr_{f_1,\ldots,f_i}\left[\Pr_x\left[\left|\Phi_{f_1,\ldots,f_{i-1},f_i^{(x)}}\right|\geq\frac{t}{\sqrt{N}}\right]>\frac{C_k}{t^{t/2}}\right]=O\left(\frac{1}{C_k t^{t/2}}\right).$$

So by the union bound,

$$\begin{aligned}\Pr_{f_1,\ldots,f_i}\left[\exists i,t:\Pr_x\left[\left|\Phi_{f_1,\ldots,f_{i-1},f_i^{(x)}}\right|\geq\frac{t}{\sqrt{N}}\right]>\frac{C_k}{t^{t/2}}\right]&=O\left(\frac{k}{C_k}\sum_{t\in[\log N]}\frac{1}{t^{t/2}}\right)\\&=O\left(\frac{k}{C_k}\right).\end{aligned}$$

■

Furthermore, good k-tuples behave as we want for property-testing purposes.

Lemma 42 *Let $\langle f_1,\ldots,f_k\rangle$ be a good k-tuple. Then for all modifications $g_1,\ldots,g_k$ such that $\Pr_x[f_i(x)\neq g_i(x)]\leq\varepsilon$ for all $i\in[k]$, we have*

$$|\Phi_{f_1,\ldots,f_k}-\Phi_{g_1,\ldots,g_k}|=O\left(k\sqrt{\varepsilon}\log\frac{1}{\varepsilon}\right).$$

Proof. Recall the "standard" quantum algorithm for k-fold FORRELATION (the one shown in Figure 1). By direct analogy to the hybrid argument of Bennett, Bernstein, Brassard, and Vazirani [7], we consider what happens if, in that algorithm, we replace the U_{f_i} oracles by U_{g_i} oracles one by one—starting with U_{f_k}, and working backwards towards U_{f_1}. Let

$$|\psi_i\rangle = \sum_{x\in\{0,1\}^n} \Phi_{f_1,\ldots,f_{i-2},f_{i-1}^{(x)}} |x\rangle$$

be the state of the quantum algorithm immediately before the i^{th} oracle call. Then by quantum-mechanical linearity, the entire sequence of oracle replacements can change the final amplitude of the all-0 state, $\alpha_{0\cdots0}$, by at most

$$\begin{aligned}
\sum_{i\in[k]} \left\| U_{f_i} |\psi_i\rangle - U_{g_i} |\psi_i\rangle \right\| &= \sum_{i\in[k]} \sqrt{\sum_{x\,:\,f_i(x)\neq g_i(x)} \left(2\Phi_{f_1,\ldots,f_{i-2},f_{i-1}^{(x)}}\right)^2} \\
&\leq 2\sum_{i\in[k]} \sqrt{\varepsilon N \left(\frac{3\log 1/\varepsilon}{\sqrt{N}}\right)^2 + \sum_{t=3\log 1/\varepsilon}^{\log N} \Pr_x\left[\left|\Phi_{f_1,\ldots,f_{i-2},f_{i-1}^{(x)}}\right| \geq \frac{t}{\sqrt{N}}\right]\left(\frac{t+1}{\sqrt{N}}\right)^2} \\
&\leq 2\sum_{i\in[k]} \sqrt{9\varepsilon\log^2\frac{1}{\varepsilon} + \sum_{t=3\log 1/\varepsilon}^{\log N} \frac{C_k}{t^{t/2}}\left(\frac{t+1}{\sqrt{N}}\right)^2} \\
&\leq 2\sum_{i\in[k]} \sqrt{9\varepsilon\log^2\frac{1}{\varepsilon} + C_k\cdot O(\varepsilon)} \\
&= O\left(k\sqrt{\varepsilon}\log\frac{1}{\varepsilon}\right).
\end{aligned}$$

But since $\alpha_{0\cdots0}$ is precisely equal to $\Phi_{f_1,\ldots,f_k}$, this means that $\Phi_{f_1,\ldots,f_k}$ can change by at most $O\left(k\sqrt{\varepsilon}\log\frac{1}{\varepsilon}\right)$ as well. ■

In contrapositive form, Lemma 42 implies that if $\Phi_{g_1,\ldots,g_k} \geq \frac{3}{5}$, then for every good k-tuple $\langle f_1,\ldots,f_k\rangle$ with $|\Phi_{f_1,\ldots,f_k}| \leq \frac{1}{100}$, there must be an $i \in [k]$ such that g_i differs from f_i on a $\Omega\left(\frac{1}{k^2\log^2 k}\right)$ fraction of points.

We now put everything together. Recall the property-testing problem, of distinguishing $\mathcal{Y}$ (the set of all $\langle f_1,\ldots,f_k\rangle$ such that $\Phi_{f_1,\ldots,f_k} \leq \frac{1}{100}$) from $\mathcal{N}_\varepsilon$ (the set of all $\langle g_1,\ldots,g_k\rangle$ that are at least ε away from $\mathcal{Y}$ in the Hamming distance sense). Lemma 40 implies that, just by taking the quantum algorithm of Proposition 6 and amplifying it a suitable number of times, we can solve this problem, with error probability at most (say) 1/3, using only $O(1/\varepsilon)$ quantum queries.[19] Furthermore, if we only need to distinguish $\mathcal{Y}$ from $\mathcal{N}_\varepsilon$ with $\Theta(\varepsilon)$ bias, then it suffices to make just 1 quantum query.

On the other hand, suppose we set $\varepsilon \leq \frac{C}{k^3\log^2 k}$, for some suitably small constant C. Then Lemma 42 implies that, if we had a randomized algorithm to distinguish $\mathcal{Y}$ from $\mathcal{N}_\varepsilon$ with constant bias, then we could also use that algorithm to distinguish the good tuples $\langle f_1,\ldots,f_k\rangle$ with $|\Phi_{f_1,\ldots,f_k}| \leq \frac{1}{100}$ from the tuples $\langle g_1,\ldots,g_k\rangle$ with $\Phi_{g_1,\ldots,g_k} \geq \frac{3}{5}$. But such an algorithm would solve

[19] Naïve repetition would give $O\left(1/\varepsilon^2\right)$ queries, but we can get down to $O(1/\varepsilon)$ using amplitude amplification.

the distributional version of k-fold FORRELATION, and we already showed (in Theorems 1 and 39 respectively) that any such algorithm requires $\Omega(\frac{\sqrt{N}}{\log N})$ queries for $k = 2$ or $\Omega(\frac{\sqrt{N}}{\log^{7/2} N})$ queries for $k > 2$.

ECCC ISSN 1433-8092

http://eccc.hpi-web.de