Separating Quantum and Classical Advice with Good Codes

John Bostanci[1], Andrew Huang[2], and Vinod Vaikuntanathan[2]

[1]Columbia University, *New York, NY*
[2]Massachusetts Institute of Technology, *Cambridge, MA*

Abstract

We show an unconditional classical oracle separation between the class of languages that can be verified using a quantum proof (QMA) and the class of languages that can be verified with a classical proof (QCMA). Compared to the recent work of Bostanci, Haferkamp, Nirkhe, and Zhandry (STOC 2026), our proof is conceptually and technically simpler, and readily extends to other oracle separations. In particular, our techniques yield the *first unconditional* classical oracle separation between the class of languages that can be decided with quantum advice (BQP/qpoly) and the class of languages that can be decided with classical advice (BQP/poly), improving on the quantum oracle separation of Aaronson and Kuperberg (CCC 2007) and the classically-accessible classical oracle separation of Li, Liu, Pelecanos and Yamakawa (ITCS 2024).

Our oracles are based on the code intersection problem introduced by Yamakawa and Zhandry (FOCS 2022), combined with codes that have extremely good list-recovery properties.

Contents

1 Introduction

We study the question of whether quantum proofs and advice provide more power than their classical counterparts. In the language of complexity theory, we consider whether the classes QMA and QCMA, and BQP/qpoly and BQP/poly, are distinct. These questions, first posed by [AN02] (for proofs) and [NY04] (for advice), have been long-standing open problems in the field of quantum complexity theory.

Given that unconditional separations seem out of reach today since they would imply breakthrough results like $\mathsf{P} \neq \mathsf{PSPACE}$, research attention has shifted to providing oracular evidence for their separation. The first progress towards this came from the seminal work of Aaronson and Kuperberg [AK07], who introduced the weaker notion of a *unitary*, or *quantum*, oracle as a means to separate the classes of QMA and QCMA and asked whether their result could be strengthened to a classical oracle separation. After a long series of works which demonstrated either conditional or non-standard separations [FK15, LLPY23, NN24, BDK24, Zha24, LMY25], the first unconditional classical oracle separation between QMA and QCMA appeared in the recent work of [BHNZ25].

The oracle used in the separation of [BHNZ25] (which was introduced in the paper of Zhandry [Zha24]) leverages one of the most natural quantum properties to separate quantum from classical computers: whether two functions are related by the Hadamard transform. This idea was first exploited in the work of [Aar10], who presented the "Forrelation" problem, i.e., deciding whether two functions are related by the Hadamard transform, as a candidate (oracle) separation between BQP and PH. The works of [Zha24] and [BHNZ25] lift this problem to QMA with a variant dubbed "spectral Forrelation". The separation of [BHNZ25] employs sophisticated techniques to analyze Forrelated oracles, including ideas inspired by mathematical physics and Hamiltonian learning. It is natural to wonder whether one can separate QMA from QCMA without using this heavy technical machinery.

Question 1: *Is there a conceptually simpler classical oracle separation between* QMA *and* QCMA*?*

A closely related problem to separating QMA from QCMA is that of separating BQP/qpoly, problems that can be solved with quantum advice, from BQP/poly, problems that can be solved with classical advice. In addition to their quantum oracle separation between QMA and QCMA, Aaronson and Kuperberg also gave a unitary oracle separation for BQP/qpoly and BQP/poly and asked if this too could be lifted to a classical oracle separation. While there is no formal equivalence relating advice and proof separations, one paradigm towards advice separations follows the general outline of [AK07]. Their advice separation begins with a hard quantum search problem (which is also used in their proof separation), and hides the value of a random language behind an oracle that expects to receive the answer to that quantum search problem. The classical oracle separation of [BHNZ25], being a kind of "classicalization" of the Aaronson-Kuperberg oracle, has a similar quantum search problem associated with it. However, since a classical oracle can no longer directly check the answer to a quantum search problem, obtaining a classical oracle separation between BQP/qpoly and BQP/poly seems to necessitate the use of a hard *classical search problem* instead.

A parallel line of work starting from the paper of Yamakawa and Zhandry [YZ24] studies the "code intersection" problem. The code intersection problem exploits another way in which quantum algorithms can exploit structure in functions: the ability to realize the convolution theorem as decoding in the Fourier basis (first noticed in [Reg09]). [YZ24] used this to formulate a TFNP problem that can be solved by an efficient quantum computer but not by a classical one: given a random oracle H and code $C \subseteq \Sigma^n$, find a codeword that hashes to the all zeros string. This problem was later modified in the works of [Liu23, LLPY23, BDK24] to make some progress towards a full separation between QMA and QCMA by restricting/modifying the query model in question (i.e. by enforcing classical-only access or bounded adaptivity). Crucially, unlike the spectral Forrelation oracle, the code intersection problem comes with a classical search problem, which [LLPY23] uses to lift their "classically-accessible" oracle search problem to get a similar (non-standard) separation between BQP/qpoly and BQP/poly. This leaves open the following question:

Question 2: *Is there a (standard) classical oracle separation between* BQP/qpoly *and* BQP/poly*?*

1.1 Our Work

We show that both these questions can be resolved using the "code intersection problem" studied in the work of Yamakawa and Zhandry [YZ24], combined with codes that have extremely good list-recovery properties. An important part of showing verifiable quantum advantage in [YZ24] was ruling out randomized classical algorithms for solving their problem. At a high level, a classical algorithm might be able to find a few symbols that

hash to zero, but it has no way of knowing whether they can be combined to form a good codeword until it tries almost every combination. As long as most (small) sets of symbols overlap with very few codewords, the chance that a classical algorithm stumbles upon a codeword which hashes to the right value is exceedingly unlikely.

One would expect that for a truly random code, no large collection of codewords can have large overlap with a small set of symbols. Inspired by the idea of pseudorandomness presented by [LMY25, Zha24], one might hope to find linear codes that mimic the properties of a random code, in this case, codes with extremely good list-recoverability. We formalize this intuition by showing that by substituting folded Reed-Solomon (FRS) codes in the construction of [YZ24] with multiplicity codes (although any code with good expansion properties would suffice), we can indeed separate QMA from QCMA.

Theorem 1.1 (Informal)**.** *There exists a classical oracle $\mathcal{O}$ such that* $\mathsf{QMA}^{\mathcal{O}} \not\subseteq \mathsf{QCMA}^{\mathcal{O}}$.

Our separation, in addition to admitting a much simpler proof, has an additional benefit over the result of [BHNZ25]: our more structured oracle problem is naturally associated with a TFNP problem, allowing us to lift our result to provide the **first unconditional classical oracle separation** between BQP/qpoly from BQP/poly in a straightforward manner.[1]

Theorem 1.2 (Informal)**.** *There exists a classical oracle $\mathcal{O}$ such that* $\mathsf{BQP}^{\mathcal{O}}/\mathrm{qpoly} \not\subseteq \mathsf{BQP}^{\mathcal{O}}/\mathrm{poly}$.

In fact, our oracle separates $\mathsf{NP}^{\mathcal{O}} \cap \mathsf{coNP}^{\mathcal{O}} \cap \mathsf{BQP}^{\mathcal{O}}/\mathrm{qpoly}$ from $\mathsf{BQP}^{\mathcal{O}}/\mathrm{poly}$, which seems to indicate a structural difference between our oracle and separations based on spectral Forrelation [Zha24, BHNZ25] or expander mixing [Lut11, NN24, LMY25], which feel like they originate from QMA-complete problems.

2 Technical Overview

The Yamakawa-Zhandry Algorithm. Our separation begins with the code intersection problem [YZ24]: given a code $C \subseteq \Sigma^n = (\mathbb{F}_q^s)^n$, function $H : [n] \times \Sigma \to \{0,1\}$, and hash $x \in \{0,1\}^n$, find a codeword $c \in C$ such that $H(i, c_i) = x_i$ for all $i \in [n]$ (we will use the shorthand $H(c) = x$ to refer to this constraint). Yamakawa and Zhandry show that when given oracle access to H, this problem has an efficient quantum algorithm but no (uniform) classical ones, giving a relativized separation between FP and FBQP. As our result will require modifying the Yamakawa-Zhandry algorithm, we begin by briefly explaining how it works.

We begin by noting that it suffices to be able to produce the state

$$|\psi\rangle \propto \sum_{v : v \in C \text{ and } H(v)=x} |v\rangle = \sum_{v \in \Sigma^n} \mathbb{1}_{H,x}(v) \cdot \mathbb{1}_C(v) \, |v\rangle \,,$$

where $\mathbb{1}_{H,x}(\cdot)$ and $\mathbb{1}_C(\cdot)$ are indicator functions for the event $H(v) = x$ and for the event $v \in C$, respectively. Taking inspiration from Regev's reduction from SIS to LWE [Reg09], Yamakawa and Zhandry observe that $|\psi\rangle$ is the pointwise product of the states

$$|\phi_1\rangle := |\phi_1(x)\rangle \propto \sum_{\mathbf{v} \in \Sigma^n : H(\mathbf{v})=x} |\mathbf{v}\rangle = \bigotimes_{i=1}^{n} \sum_{v_i \in \Sigma : H(i,v_i)=x_i} |v_i\rangle \qquad \text{and} \qquad |\phi_2\rangle \propto \sum_{\mathbf{v} \in C} |\mathbf{v}\rangle \,,$$

both of which can be prepared efficiently given access to H. By the convolution theorem, we know that

$$\mathrm{QFT}_q \, |\psi\rangle \propto \mathrm{QFT}_q(|\phi_1\rangle \odot |\phi_2\rangle) = \mathrm{QFT}_q \, |\phi_1\rangle \star \mathrm{QFT}_q \, |\phi_2\rangle \,,$$

where $\odot$ denotes the point-wise product of two vectors, and $\star$ their convolution. Therefore, it suffices to efficiently prepare the state $|\mathsf{goal}\rangle := \mathrm{QFT}_q \, |\phi_1\rangle \star \mathrm{QFT}_q \, |\phi_2\rangle$, as $\mathrm{QFT}_q^{-1} \, |\mathsf{goal}\rangle = |\psi\rangle$. If C is a $\mathbb{F}_q$-linear code, then $\mathrm{QFT}_q \, |\phi_2\rangle$ is simply the uniform superposition over the dual code $C^\perp$, so we can produce the states

$$\mathrm{QFT}_q \, |\phi_1\rangle \otimes \mathrm{QFT}_q \, |\phi_2\rangle \propto \sum_{\mathbf{e} \in \Sigma^n} \sqrt{\mathcal{D}_{H,x}(\mathbf{e})} \, |\mathbf{e}\rangle \otimes \sum_{\mathbf{v} \in C^\perp} |\mathbf{v}\rangle$$

[1] A similar idea appeared in [LLPY23], where a variant of the Yamakawa-Zhandry problem was used to give a separation between quantum and classical advice for algorithms which are only allowed classical access to all oracles. As we will see later, our separation strictly improves on this result since we rule out all BQP/poly algorithms with *quantum* oracle access while only requiring a single classical query given quantum advice.

$$\overset{U_{\mathsf{add}}}{\mapsto} \sum_{\mathbf{e}\in\Sigma^n,\mathbf{v}\in C^\perp} \sqrt{\mathcal{D}_{H,x}(\mathbf{e})}\,|\mathbf{e}\rangle\,|\mathbf{v}\rangle\,|\mathbf{v}+\mathbf{e}\rangle \overset{U_{\mathsf{sub}}}{\mapsto} \sum_{\mathbf{e}\in\Sigma^n,\mathbf{v}\in C^\perp} \sqrt{\mathcal{D}_{H,x}(\mathbf{e})}\,|0\rangle\,|\mathbf{v}\rangle\,|\mathbf{v}+\mathbf{e}\rangle\,,$$

while our desired state is

$$|\mathsf{goal}\rangle = \mathrm{QFT}_q\,|\phi_1\rangle \star \mathrm{QFT}_q\,|\phi_2\rangle \propto \sum_{\mathbf{v}\in C^\perp,\mathbf{e}\in\Sigma^n} \sqrt{\mathcal{D}_{H,x}(\mathbf{e})}\,|\mathbf{v}+\mathbf{e}\rangle\,,$$

where $\mathcal{D}_{H,x}(\cdot)$ is the density function of $\mathrm{QFT}_q\,|\phi_1\rangle$ (here, we are ignoring phases for simplicity of exposition). For a completely random function H and a fixed x, we observe that $\mathcal{D}_{H,x}(\cdot)$ will have roughly half of its weight on 0 in each coordinate (since around half of all symbols in each coordinate should hash to 0 or 1), while the remaining half of its weight will be close to uniform over nonzero symbols. By taking C to be a folded Reed-Solomon (FRS) code with sufficiently high rate, Yamakawa and Zhandry show that one can efficiently decode $C^\perp$ from errors over $\mathcal{D}_{H,x}(\cdot)$ with high probability and consequently prepare $|\mathsf{goal}\rangle$ as desired.

Lifting the Separation. The obvious issue in using the code intersection problem to separate QMA from QCMA, however, is that both QMA and QCMA algorithms can make quantum oracle access to H, and our problem is already in BQP! Thus, we must make some modifications to the problem at hand.

First, as observed by [Liu23], the first phase of this algorithm can be made non-adaptive: the state $|\phi_2\rangle$ depends only on C, while $|\phi_1\rangle$ depends only mildly on x, since we can simply prepare all $2n$ preimage states

$$\left|H_1^{-1}(0)\right\rangle := \sum_{x\in\Sigma:H(1,x)=0} |x\rangle\,,\ldots,\left|H_n^{-1}(1)\right\rangle := \sum_{x\in\Sigma:H(n,x)=1} |x\rangle$$

before selecting $|\phi_1\rangle$ when given x. On the other hand, the second phase does not require access to H, and should work equally well for all x. Therefore, if we are given the state

$$|\mathsf{adv}_H\rangle := \bigotimes_{i=1}^n \left|H_i^{-1}(0)\right\rangle \otimes \bigotimes_{i=1}^n \left|H_i^{-1}(1)\right\rangle \otimes |\phi_2\rangle$$

as our quantum proof or advice, we can produce $|\phi_1(x)\rangle := \bigotimes_{i=1}^n \left|H_i^{-1}(x_i)\right\rangle$ for *any* x and use it to produce a solution to the code intersection problem for x without access to H! We can therefore replace H with the much weaker oracle $O_H(x,v)$ that simply verifies if the vector v hashes to x.

Of course, this problem is still in $\mathsf{NP}\subseteq\mathsf{QCMA}$, since for a fixed x, the prover can always send any codeword v that hashes to x! Luckily, we claim that this is pretty much the *only* thing that the prover can do. To operationalize this intuition, we note that our quantum proof/advice is in some sense encoding many codewords along with their hashes in superposition. Thus, taking some inspiration from [LLPY23, BDK24], we define the *Code Intersection Subset Size* problem as follows:

> Estimate the size of a set $E\subseteq\{0,1\}^n\times\Sigma^n$, which is promised to either be the full set $\{0,1\}^n\times\Sigma^n$ or a small subset $E\subseteq F\times\Sigma^n\subseteq\{0,1\}^n\times\Sigma^n$ where $|F|\leq t$ for some threshold $t\ll 2^n$, given access to the following oracle $O[H,E](x,v)$:
>
> $$O[H,E](x,v) = \begin{cases} 1 & \text{if } v\in C,\ H(v)=x, \text{ and } (x,v)\in E,\\ 0 & \text{otherwise.}\end{cases}$$

This problem is naturally in QMA: the prover can give as proof $|\mathsf{adv}_H\rangle$ which depends only on H. Given $|\mathsf{adv}_H\rangle$, the verifier can sample a random $x\in\{0,1\}^n$ and should be able to produce with high probability $v\in C$ such that $H(v)=x$. The oracle O therefore allows the verifier to check if $x\in E$ for any x, making the set estimation problem trivial. In fact, the QMA verifier need not be concerned about malicious proofs, since a NO instance of O always outputs 0 whenever x (which is sampled solely by the verifier) is not in F.

We note that our problem differs from the oracle problems defined in [LLPY23, BDK24], which take $E=F\times\Sigma^n$ for some small set F. This modification, while not impacting our QMA algorithm, will be crucial in establishing a QCMA lower bound.

An Entropic Viewpoint. To rule out QCMA proof systems, we first observe that the major difference between quantum and classical proofs lies in their *clonability*. In particular, an (oracle) algorithm which uses a *classical* witness can always be re-run with the same witness, even if it makes measurements. This simple and seemingly obvious fact, first formally identified in [Zha24], was utilized to great effect by [BHNZ25] to give their classical oracle separation, and we will take advantage of it as well.

To this end, suppose there was some Q-query QCMA verifier V which succeeded in the set estimation problem. We observe that this means that V can always distinguish between $O[H, \{0,1\}^n \times \Sigma^n]$ and $O[H, E]$ whenever E is small. But these oracles differ only at inputs (x, v) where $v \in C$, $H(v) = x$, and $(x, v) \notin E$! Thus, by the hybrid lemma [BBBV97], if we measure a random query that V makes to $O[H, E]$, we should expect to get a new pair $(x, v) \notin E$ such that $v \in C$ and $H(v) = x$ with good probability provided E is small.[2] This gives rise to a natural algorithm for *guessing* the hash values of codewords: starting with $E = \emptyset$, simulate a run of V with $O[H, E]$ and measure a random query before adding the measurement outcome to E; rinse and repeat. By our previous argument, conditioned on having a good witness, each iteration of this algorithm should correctly produce a new codeword and hash with non-negligible probability. We can therefore turn V into a guesser with non-uniform advice which correctly produces many distinct codewords and their hash values without making *any* oracle queries. After guessing the classical w-bit witness, this gives rise to an unconditional no-query algorithm which guesses the hash values of ℓ codewords in C for all $\ell \leq t$ with probability

$$2^{-w} \cdot \left(\Omega(Q^{-2})\right)^{\ell} = 2^{-\mathrm{poly}(n)} \cdot \left(\frac{1}{\mathrm{poly}(n)}\right)^{\ell}.$$

We now argue that this is in fact impossible. Observing that H is independently random at each coordinate i and symbol $x \in \Sigma$, we can upper bound the success probability of any sampler which produces ℓ points by $(\frac{1}{2})^{s(\ell)}$, where $s(\ell)$ is the minimum number of symbols that appear among ℓ distinct codewords in C. Taking $\ell = t = \omega(\mathrm{poly}(n))$, if we can argue that $s(\ell) = \omega(\log n \cdot \ell)$, then we see that

$$2^{-\mathrm{poly}(n)} \cdot \left(\frac{1}{\mathrm{poly}(n)}\right)^{\ell} = \left(\frac{1}{\mathrm{poly}(n)}\right)^{\ell} \gg \left(\frac{1}{2^{\omega(\log n)}}\right)^{\ell} = \left(\frac{1}{2}\right)^{s(\ell)},$$

which will give us our desired contradiction.

List Recovery and Code Expansion. How might we bound $s(\ell)$? For any ℓ distinct codewords $c_1, \ldots, c_\ell \in C$, define the lists $S_1, \ldots, S_n \subseteq \Sigma$ such that S_i consists of all symbols in Σ that appear in the i'th coordinate of some codeword c_j for $j \in [\ell]$. Clearly, $s(\ell) = \min_{c_1,\ldots,c_\ell} \sum_{i=1}^{n} |S_i|$, so there must be lists $S_1^*, \ldots, S_n^*$ such that

$$|C \cap (S_1^* \times \ldots \times S_n^*)| \geq \ell \qquad \text{and} \qquad \sum_{i=1}^{n} |S_i^*| = s(\ell).$$

We now see that the question of how small $s(\ell)$ can be is precisely characterized by the *list-recoverability* of C. In particular, if we know that C is $(L, O(L))$-list-recoverable for $L \lesssim \ell$, then this would mean that $\frac{s(\ell)}{n} = \frac{1}{n}\sum_{i=1}^{n} |S_i^*| = \Omega(\ell)$ as desired!

Note that this is a pretty strong condition; it necessitates the use of codes that have *near-optimal list recovery*, a property that in particular is not satisfied by the FRS codes used by [YZ24] (see Subsection §3.3 for further discussion). Fortunately, there is a fix: the setting of zero-error list recovery is closely linked to the notion of unbalanced expanders, and the recent work of [KTS22] shows that multiplicity codes exhibit precisely the sort of list recovery that we need. Moreover, the fact that multiplicity codes are $\mathbb{F}_q$-linear and that their duals have relatively good distance [RZVW24] should guarantee the success of our QMA algorithm. We note that although our dual code happens to admit efficient unique decoding, our separation only needs $C^\perp$ to be *combinatorially* uniquely decodable, since we can always provide an (inefficient) decoding oracle.

[2] Critically, if we have already successfully guessed some collection E of code words and hash values, we can always perfectly simulate $O[H, E]$. The same is *not* true for the oracles of [LLPY23, BDK24], for which even "small" sets correspond to exponentially many codewords.

A Final Complication. It seems that this rather simple argument completes our separation; after all, by switching to using multiplicity codes (rather than folded Reed-Solomon codes as in [YZ24, LLPY23, BDK24]), we have been able to rule out all possible QCMA algorithms. Sadly, we have to deal with one final and rather subtle issue, which has to do with the parameters of the multiplicity codes: in the process of obtaining excellent list recovery/expansion from our multiplicity codes, we are forced to make the relative rate of our (primary) code sub-constant, which means our dual code now has sub-constant relative distance! Recalling that our error distribution should concentrate on vectors with Hamming weight roughly $n/2$, we observe that this level of noise is now likely intolerable as there may not even exist a unique decoding most of the time under this error distribution.

Our solution is relatively simple, and it uses the generous amount of flexibility that the [Reg09, YZ24] algorithm affords us. Instead of using a completely random function H, we will instead make our function *biased* in favor of 0 (reminiscent of a recent strategy employed by [GGJL25]). That is, for each element $\sigma \in \Sigma$, $H(i, \sigma)$ will take on the value 0 with probability $p \gg \frac{1}{2}$. Thus, for $x = 0^n$, we can expect the error vectors in $\mathrm{QFT}\,|\phi_1\rangle$ to have Hamming weight $\approx n(1-p)$, drastically reducing the amount of noise that we are required to decode from with respect to $C^\perp$.

This change does not come for free, however: unlike in [GGJL25], where the goal was to invert only $x = 0^n$, we need to be able to invert many x's, including those with large Hamming weight. By biasing H towards 0, on inputs like $x = 1^n$, we create an error distribution which has expected Hamming weight $\approx np \gg n/2$, thereby *worsening* our ability to invert!

Our final idea is to observe that since our algorithm can only invert low Hamming-weight vectors x rather than all vectors in $\{0,1\}^n$, we can simply *modify the problem to enforce this condition.* Instead of trying to invert all $x \in \{0,1\}^n$, we can focus on inverting vectors of the form $x\|0^{n-n^c}$, where $x \in \{0,1\}^{n^c}$ and $0 < c \ll 1$. That is, we will now try to differentiate $E = \{0,1\}^{n^c} \times 0^{n-n^c} \times \Sigma^n$ from $E \subseteq F \times 0^{n-n^c} \times \Sigma^n$ where $|F| \leq t \ll 2^{n^c}$. Since $x\|0^{n-n^c}$ has Hamming weight at most n^c, the corresponding error distribution for $|\phi_1(x\|0^{n-n^c})\rangle$ will be concentrated on vectors with Hamming weight at most $n^c + (n-n^c)(1-p) \approx n(1-p)$, guaranteeing the success of our QMA algorithm provided our dual code has distance at least $O(n(1-p))$ and $p \lesssim 1 - 1/n^{1-c}$.

A General Recipe for a Classical Oracle Separation. Before describing the advice separation, we summarize all of the steps we have taken so far to provide a general recipe for getting a QMA versus QCMA oracle separation. We start with an infinite family of codes $\{C_\lambda\}_{\lambda\in\mathbb{N}}$ over a large alphabet Σ (so that there are many solutions to the code intersection problem) such that:

1. $C_\lambda^\perp$ can be efficiently decoded up to $\Omega(\lambda^{1+c})$ errors for any constant c,
2. Codewords of C_λ consist of $n = \mathrm{poly}(\lambda)$ many symbols from Σ,
3. C_λ has near optimal list recovery for sufficiently large λ.

Then for every λ, we can sample hash functions $H_1, \ldots, H_n$ to be biased so that roughly a $\lambda/n(\lambda)$-fraction of symbols are pre-images of 1. We will ask for pre-images of $x\|0^{n-\lambda}$ for $x \in \{0,1\}^\lambda$. The bias of the H_i, together with the fact that we ask for a hash that has at most λ many 1's, ensures that the dual decoding problem encounters an error with $O(\lambda)$ Hamming weight with high probability, which falls under our dual decoding distance of $\Omega(\lambda^{1+c})$. Thus, the Yamakawa-Zhandry algorithm works and the problem stays in QMA. At the same time, a QCMA verifier will imply a sampler that outputs v codewords of the code with probability $\mathrm{poly}(\lambda)^{-v}$, and list recovery will enforce that this corresponds to $\Omega(v \cdot n)$ symbols. The bias of the H_i will mean that the probability of guessing all symbols correctly will be $(1-\lambda/n)^{\Omega(v\cdot n)} \approx \exp(-\lambda v) \ll \mathrm{poly}(\lambda)^{-v}$, giving us a contradiction.

By instantiating this recipe with carefully chosen multiplicity codes, we arrive at our classical oracle separation.

Moving to the Advice Setting: BQP/qpoly vs. BQP/poly. Unfortunately, the set approximation problem is easy with *trusted* advice: a single classical bit suffices to indicate whether the set in question is large or small. Here, the relative simplicity of our separation (which uses almost entirely classical ideas) and our use of a TFNP problem allows us to extend our separation to the question of BQP/qpoly and BQP/poly. In contrast, it is not at all obvious (to us) how to construct even a candidate separating language based on the (decisional) spectral Forrelation problem of [BHNZ25].

In particular, the code intersection problem gives rise to the following BQP/qpoly language: begin by sampling some random binary language $\mathcal{L} \subseteq \{0,1\}^n$. On input $x \in \{0,1\}^n$, our oracle O will simply return whether

$x \in \mathcal{L}$ or $x \notin \mathcal{L}$ provided it is given a valid codeword v which hashes to x. On the one hand, our original QMA algorithm still works as a BQP/qpoly machine, since $|\mathsf{adv}_H\rangle$ is agnostic of x and allows us to produce v for all x. On the other hand, as we have intuitively argued earlier, even trusted classical advice should not help a BQP machine to produce many valid codewords, so for most x, any BQP machine with classical advice will not be able to receive the output of the oracle indicating whether $x \in \mathcal{L}$. Since the advice is bounded, it cannot itself describe many elements of $\mathcal{L}$, so the BQP machine will fail to decide whether $x \in \mathcal{L}$ for most x. The problem of guessing the value of a random function H given bounded-size advice and without querying H can be made precise by appealing to results on Yao's box problem due to Chung, Guo, Liu and Qian [CGLQ20]. We show that relatively straightforward modifications of the argument used by [LLPY23] for ruling out *classical-access* BQP/poly algorithms extend to ruling out generic BQP/poly algorithms as well, completing the advice separation.

Some Concluding Remarks. First, as in [BHNZ25], it is not hard to see that our oracles also separate the clonable variants of QMA and BQP/qpoly from their regular counterparts.

Secondly, we emphasize that while the code intersection problem has previously been considered in the context of proof and advice separations [Liu23, LLPY23, BDK24], all existing works employed FRS codes, which (as mentioned earlier) do not enjoy strong enough list-recovery properties. In particular, running our argument with FRS codes gives an upper bound on the sampling success probability which is too weak for *any* (even weak) separation! On the other hand, multiplicity codes do not appear to have strong enough decoding properties to handle the large amounts of noise that would be incurred by perfectly random functions H, which necessitates our use of biased oracles and restriction to low Hamming-weight vectors (an idea which did not appear in earlier, more limited, separations).

Finally, and unrelated to the main theme of this paper, in the process of modifying the Yamakawa-Zhandry algorithm for our separation, we observe that biasing the random oracle H in question also has the benefit of making the quantum algorithm in question *much more efficient*. As an example, if we tweak H to be even 2/3-biased, this already decreases the noise level sufficiently that we can rely on (significantly faster) *unique decoders* rather than list decoders, which currently appears to be the major bottleneck in runtime.

3 Discussion and Open Questions

3.1 Structured versus randomness in classical oracle separations

As stated in the introduction, one major barrier in lifting the oracle separation of [BHNZ25] to the advice setting is that the oracle is most naturally associated with a hard quantum search problem, instead of a classical one. In their classical oracle separation, the authors identify a way of sampling instances of the spectral Forrelation problem such that the pair of functions that seems completely random, except that they are related by the Fourier transform (and, in the case of [BHNZ25], one of the two oracles being sparse). In some informal sense, our oracle separation enables an efficient quantum verifier to extract more information (namely, the solution to a hard search problem) from its witness, but doing so seems to require additional structure in the oracle.

A natural question to ask is how much information can be encoded into a quantum state before it becomes clonable, and whether our ideas are useful in encoding information into other kinds of quantum states. The cryptographic analogy of a separation between QMA and QCMA (or, really UnclonableQMA) is a primitive called quantum money [Aar09]. These are states that can not be cloned, but can be verified, similar to witnesses for QMA problems that are not in QCMA. An extremal form of this primitive (and this idea of encoding information in a quantum state) is known as "copy-protected software", wherein an efficient quantum party can extract the input output behavior of an entire classical function from a quantum state. Currently, there are several candidates for quantum money constructions in the plain model [FGH+12, BNZ25, Zha25], but less is known about copy-protected software. We hope that ideas from our separation might be useful in finding such constructions. To make progress towards cryptographic instantiations, one concrete direction to explore is a precise characterization of which witnesses cause our QMA verifier to accept with high probability.

3.2 QMA-completeness of a decoding problem

One interesting difference between our oracle separation and the oracle separation of [BHNZ25] is that their oracle separation can be seen as an obfuscation of a QMA-complete problem. To elaborate further, just as problems

involving random sparse functions might model the difficulty of constructing a SAT solver which ignores the structure of the SAT instance it receives, the spectral Forrelation problem models an algorithm for solving a two-basis local Hamiltonian problem that does not look at the structure of the two local Hamiltonians it receives. Note that despite the connection to a QMA-complete problem, this property is not actually needed to achieve a separation between QMA and QCMA, as highlighted in the actual oracle separation of [BHNZ25]. In particular, the "YES" and "NO" instances can be taken to have sets S of a fixed size ℓ or $\leq \ell/10$. Such *distributions* of oracles can be easily distinguished by an AM protocol [GS86], but it appears unlikely that solving the actual spectral Forrelation problem can be done in AM, because another way to sample "NO" instances of the spectral Forrelation problem would be to re-sample sets S' independent of U. Distinguishing such pairs from Forrelated pairs (S, U) seems to truly require a QMA verifier.

In contrast, the problem we construct really is in AM, as it directly involves distinguishing between large and small sets, and any reasonable variant of the code intersection problem would likely remain in the polynomial hierarchy. Of course, an oracle separation between QCMA and AM already exists (and in fact, comes from early work on separating QMA from QCMA [FK15]!), but we find it intriguing that our separation relies on a problem which appears to be of only intermediate difficulty and does not need the "full" power of QMA in some sense. In fact, as our problem appears on its face to be completely unrelated to quantum algorithms, we believe it remains an extremely interesting question to find a QMA-complete variant of the code intersection problem.

3.3 Simplifications to the separation

Naturally, one may ask if our separations can be made *even simpler*; here, we outline a few directions to consider.

On Round-Reduction Arguments. The work of [BDK24] can be thought of as a round-reduction argument as follows: in the style of [AK07], we will, given any QCMA proof system, fix some classical witness which corresponds to the largest set of NO instances. The polynomial bound on the size of the proof means that the collection of functions H which are consistent with this witness remains substantial, and in particular the distribution over all consistent H must have large min-entropy. We conclude that the set of symbols S which have low entropy conditioned on this classical proof is also bounded.

We now look at the verifier's inputs to the oracle in the first round of queries. Observe that any inputs corresponding to non-codewords, as well as ones which have low overlap with S will be correct with negligible probability. By the hybrid lemma, it suffices to restrict our attention to codewords which have high overlap with S (so called "dangerous" inputs) – but this set is bounded precisely by the list recovery of our code! At this stage, we can simply give away the values of H on dangerous inputs *for free*, allowing the verifier to simulate its first round of queries. As a consequence, we can peel off a round of queries at the cost of requiring more advice. Repeating this "peeling and bloating" routine with FRS codes results in a $o(\log n/\log\log n)$-round bound.

Sadly, this approach seems fairly doomed if we stick to the Yamakawa-Zhandry problem: even with nearly optimal list recovery, our advice will certainly increase by some constant factor with each round, so we would remain stuck at an $o(n)$-round bound. One could imagine a better argument that uses some special property of any collection of recovered codewords that ensures these lists do not grow by too much iteratively, but this seems quite difficult (and would definitely be much more complicated!).

Other Codes? One might wonder why we need multiplicity codes here – after all, there are other codes which give rise to unbalanced expanders besides those of [KTS22], namely the constructions of [GUV09], but examining these constructions in closer detail presents some unexpected issues:

1. [GUV09] construct two unbalanced expanders with near-optimal expansion, one based on Parvaresh-Vardy codes and the other based on a subcode of FRS codes. Alas, neither of these instantiations are linear, which means our quantum proof/advice-based algorithm will fail.

2. [GUV09] also considers unbalanced expanders based on plain FRS codes, which *are* linear in some sense, but such expanders have expansion which is too weak to show classical hardness.

On the Necessity of Near-Optimal List Recovery. Our proof relies on fairly strong expansion/list recovery properties of the underlying code C. It is not hard to see that we can tolerate slightly suboptimal expansion, i.e.

if $s(\ell) = \Omega(\ell/\text{poly}(n))$. Extending our sampling-based argument to work for polynomial or super-polynomial expansion, i.e. $s(\ell) = \ell^{O(1)}$, would open the door to a much larger class of usable codes.

What About Random Linear Codes? It is well known that random linear codes (RLCs) and their duals have good distance with high probability, and they are by definition linear, which means our quantum proof/advice-based algorithm will succeed. However, to show classical hardness, we need near-optimal list recovery, which remains a challenging open problem to show for RLCs [LS25]. We believe that a proof of such a result for RLCs is the clearest way to conceptually simplify our separation.

4 Preliminaries

4.1 Notation

We say that a function $\delta : \mathbb{N} \mapsto [0,1]$ is inverse polynomial if there exists a polynomial p such that $\delta(n) \leq 1/p(n)$ for sufficiently large n. A function $\epsilon : \mathbb{N} \mapsto [0,1]$ is negligible if for every polynomial p, for all sufficiently large n, $\epsilon(n) < 1/p(n)$.

We use the notation id to denote the identity operator. We will occasionally concatenate superscripts when it is clear from context, so $\text{QFT}^{-1,\otimes n}$ denotes $(\text{QFT}^{-1})^{\otimes n}$. We will also sometimes abbreviate the tensor product state $|0\rangle^{\otimes n}$ as $|0^n\rangle$.

A register R is a named finite-dimensional complex Hilbert space. If $\mathsf{A}, \mathsf{B}, \mathsf{C}$ are registers, for example, then the concatenation ABC denotes the tensor product of the associated Hilbert spaces. For a linear transformation L and register R, we write L_{R} to indicate that L acts on R, and similarly we write σ_R to indicate that a state σ is in the register R.

4.2 Probability and Complexity Theory

Definition 4.1 (Modified from [BHNZ25])**.** *We use the phrase "an oracle" to refer to a function $\mathcal{O} : \{0,1\}^* \to \{0,1\}$. A quantum query algorithm is a quantum circuit that interacts with an oracle $\mathcal{O} : \{0,1\}^* \to \{0,1\}$ via a query gate $|x,b\rangle \to |x, b \oplus \mathcal{O}(x)\rangle$, which acts on an $|x|+1$-qubit query register for various x. The algorithm is described by an alternating sequence of unitaries (drawn from any fixed gate set) and query gates. After all gates are applied, a designated qubit is measured in the standard basis to determine acceptance. The circuit has ancilla qubits which are initialized to $|0\rangle$ and all intermediate unitaries may act on an arbitrary (but finite) number of qubits.*

A quantum query algorithm may also receive an auxiliary witness or advice as input. A quantum witness/advice state is a state $|\psi\rangle$ on some (finite) number of qubits, while a classical witness/advice string is a (finite) bitstring w, treated as a computational basis state. The algorithm's acceptance probability may depend on both the oracle and the witness.

Definition 4.2 ([BHNZ25])**.** *We denote a family of quantum oracle circuits/algorithms by $\{\mathcal{A}_\lambda\}_{\lambda\in\mathbb{N}}$, where the index λ corresponds to the length of the explicit input to the computational problem. $\{\mathcal{A}_\lambda\}_{\lambda\in\mathbb{N}}$ is P-uniform if there exists a deterministic polynomial-time Turing machine M that, on input 1^λ, outputs a full classical description of the circuit $\mathcal{A}_\lambda$. The runtime of M implies that $\mathcal{A}_\lambda$ has at most $\text{poly}(\lambda)$ gates (oracle or elementary), queries $\mathcal{O}$ at lengths of at most $\text{poly}(\lambda)$, and receives witnesses/advice of length at most $\text{poly}(\lambda)$.*

We can now define the complexity classes that we will consider in this work.

Definition 4.3 (Oracle QCMA)**.** *A promise language $\mathcal{L}^{\mathcal{O}} = (\mathcal{L}_{\mathsf{yes}}, \mathcal{L}_{\mathsf{no}})^{\mathcal{O}} \subseteq \{0,1\}^*$ is in $\mathsf{QCMA}^{\mathcal{O}}$ if there exists a P-uniform family of quantum oracle circuits $\mathcal{A}_\lambda$ with $\mathcal{A}_\lambda$ accepting a witness of length $t(\lambda)$, such that for every input x of length $\lambda = |x|$,*

- $x \in \mathcal{L}_{\mathsf{yes}} \implies \exists w \in \{0,1\}^{t(\lambda)}$ *s.t.* $\Pr[\mathcal{A}^{\mathcal{O}}_\lambda(x,w) = 1] \geq \frac{2}{3}$,
- $x \in \mathcal{L}_{\mathsf{no}} \implies \forall \widetilde{w} \in \{0,1\}^{t(\lambda)}, \Pr[\mathcal{A}^{\mathcal{O}}_\lambda(x,\widetilde{w}) = 1] \leq \frac{1}{3}$.

Definition 4.4 (Oracle QMA)**.** *A promise language $\mathcal{L}^{\mathcal{O}} = (\mathcal{L}_{\mathsf{yes}}, \mathcal{L}_{\mathsf{no}})^{\mathcal{O}} \subseteq \{0,1\}^*$ is in $\mathsf{QMA}^{\mathcal{O}}$ if there exists a P-uniform family of quantum oracle circuits $\mathcal{A}_\lambda$ with $\mathcal{A}_\lambda$ accepting a witness of length $t(\lambda)$, such that for every input x of length $\lambda = |x|$,*

- $x \in \mathcal{L}_{\mathsf{yes}} \implies \exists\, |\psi\rangle \in (\mathbb{C}^2)^{\otimes t(\lambda)}$ *s.t.* $\Pr[\mathcal{A}^{\mathcal{O}}_\lambda(x, |\psi\rangle) = 1] \geq \frac{2}{3}$,
- $x \in \mathcal{L}_{\mathsf{no}} \implies \forall\, |\widetilde{\psi}\rangle \in (\mathbb{C}^2)^{\otimes t(\lambda)}, \Pr[\mathcal{A}^{\mathcal{O}}_\lambda(x, |\widetilde{\psi}\rangle) = 1] \leq \frac{1}{3}$.

Definition 4.5 (Oracle BQP/poly)**.** *A promise language $\mathcal{L}^{\mathcal{O}} = (\mathcal{L}_{\mathsf{yes}}, \mathcal{L}_{\mathsf{no}})^{\mathcal{O}} \subseteq \{0,1\}^*$ is in $\mathsf{BQP}^{\mathcal{O}}$/poly if there exists a P-uniform family of quantum oracle circuits $\mathcal{A}_\lambda$ with $\mathcal{A}_\lambda$ accepting advice of length $t(\lambda)$ and an advice family $\{\mathsf{adv}_\lambda\}_{\lambda \geq 1}$ where $|\mathsf{adv}_\lambda| = t(\lambda)$, such that for every input x of length $\lambda = |x|$,*

- $x \in \mathcal{L}_{\mathsf{yes}} \implies \Pr[\mathcal{A}^{\mathcal{O}}_\lambda(x, \mathsf{adv}_\lambda) = 1] \geq \frac{2}{3}$,
- $x \in \mathcal{L}_{\mathsf{no}} \implies \Pr[\mathcal{A}^{\mathcal{O}}_\lambda(x, \mathsf{adv}_\lambda) = 1] \leq \frac{1}{3}$.

If $t(\lambda) = 0$ then we say $\mathcal{L}^{\mathcal{O}} \in \mathsf{BQP}^{\mathcal{O}}$.

Definition 4.6 (Oracle BQP/qpoly)**.** *A promise language $\mathcal{L}^{\mathcal{O}} = (\mathcal{L}_{\mathsf{yes}}, \mathcal{L}_{\mathsf{no}})^{\mathcal{O}} \subseteq \{0,1\}^*$ is in $\mathsf{BQP}^{\mathcal{O}}$/qpoly if there exists a P-uniform family of quantum oracle circuits $\mathcal{A}_\lambda$ with $\mathcal{A}_\lambda$ accepting advice of length $t(\lambda)$ and an advice family $\{|\mathsf{adv}_\lambda\rangle\}_{\lambda \geq 1}$ where $|\mathsf{adv}_\lambda\rangle \in (\mathbb{C}^2)^{\otimes t(\lambda)}$, such that for every input x of length $\lambda = |x|$,*

- $x \in \mathcal{L}_{\mathsf{yes}} \implies \Pr[\mathcal{A}^{\mathcal{O}}_\lambda(x, |\mathsf{adv}_\lambda\rangle) = 1] \geq \frac{2}{3}$,
- $x \in \mathcal{L}_{\mathsf{no}} \implies \Pr[\mathcal{A}^{\mathcal{O}}_\lambda(x, |\mathsf{adv}_\lambda\rangle) = 1] \leq \frac{1}{3}$.

Given a quantum query algorithm, we can define the query mass of the algorithm on a particular set of inputs.

Definition 4.7 (Query mass)**.** *For an oracle circuit A making Q quantum queries to an oracle $\mathcal{O}$ with input domain D, let $\sum_x \alpha_x^{(i)} |x\rangle |\psi_x\rangle$ be the state of the algorithm immediately before their i'th query to $\mathcal{O}$, where the first register is the input register to the oracle, and let $M_x(i) = |\alpha_x^{(i)}|^2$ be the* query mass *of x in the i'th query. For a subset $V \subseteq [Q] \times D$, let $M_V = \sum_{(i,x) \in V} M_x(i)$ be the* total query mass *of points in V.*

The following theorem was proven in [BBBV97], using the hybrid method.

Theorem 4.8 (Hybrid method [BBBV97])**.** *Let A be an oracle circuit which makes Q queries to an oracle $\mathcal{O}$ with input domain D. If we modify $\mathcal{O}$ into an oracle $\mathcal{O}'$ which differs only on a set of time-input pairs $V \subseteq [Q] \times D$, then*

$$|\Pr[A^{\mathcal{O}}(\cdot) = 1] - \Pr[A^{\mathcal{O}'}(\cdot) = 1]| \leq 4\sqrt{QM_V}\,.$$

Finally, we will also use some basic probability lemmas.

Lemma 4.9 (Chernoff Bound)**.** *Let $X_1, \ldots, X_n$ be independent random variables taking values in $\{0,1\}$, $X := \sum_{i=1}^n X_i$, and $\mu := \mathbb{E}[X]$. For any $\delta \geq 0$, it holds that $\Pr[X \geq (1+\delta)\mu] \leq e^{-\delta^2 \mu/(2+\delta)}$.*

Lemma 4.10 (Borel–Cantelli, [Bor09, Can17])**.** *Let $\{X_\lambda\}_{\lambda \in \mathbb{N}}$ be a sequence of (not necessarily independent) random variables with values in $\{0,1\}$. If $\sum_{\lambda=1}^\infty \mathbb{E}[X_\lambda] < \infty$, then $\Pr\left[\sum_{\lambda=1}^\infty X_\lambda = \infty\right] = 0$.*

4.3 Coding Theory

For a prime power q, we denote by $\mathbb{F}_q$ the finite field of order q and denote by $(\mathbb{F}_q)_{<k}[X]$ the set of univariate polynomials over $\mathbb{F}_q$ with degree less than k.

Definition 4.11. *A code of length $n \in \mathbb{N}$ over an alphabet Σ is a subset $C \subseteq \Sigma^n$. $C \subseteq \Sigma^n$ is said to be $\mathbb{F}_q$-*linear *if its alphabet $\Sigma = \mathbb{F}_q^s$ for some field $\mathbb{F}_q$ and a positive integer $s \geq 1$ and C is an $\mathbb{F}_q$-linear subspace of Σ^n. Equivalently, this means that for any two codewords $x, y \in C$ and scalar $\alpha \in \mathbb{F}_q$, both $x + y$ and $\alpha \cdot x$ are in C.*

For an $\mathbb{F}_q$-linear code C, the dual code of C is the code $C^\perp \subseteq (\mathbb{F}_q^s)^n$ containing all strings $c' \in (\mathbb{F}_q^s)^n$ which satisfy

$$\sum_{i=1}^n \sum_{j=1}^s (c'_i)_j \cdot (c_i)_j = 0$$

for all $c \in C$. Observe that $C^\perp$ is always $\mathbb{F}_q$-linear, and that $|C| \cdot |C^\perp| = |\Sigma|^n$ if and only if C is $\mathbb{F}_q$-linear.

For any vector $x \in \Sigma^n$, define $\mathsf{hw}(x) \in [0, n]$ as the Hamming weight of x, i.e. the number of nonzero elements in x. We say that $C \subseteq \Sigma^n$ has distance d if for any two distinct codewords $c_1, c_2 \in C$, $\mathsf{hw}(c_1 - c_2) \geq d$.

Definition 4.12 (Formal and Hasse derivatives)**.** *Let $f(X) = \sum_{j=0}^{n} a_j X^j \in \mathbb{F}_q[X]$ be a univariate polynomial over $\mathbb{F}_q$. We define the i'th formal and Hasse derivatives of $f(X)$ as the linear operators which take $f(X)$ to the polynomials*

$$f^{[i]}(X) = \sum_{j=i}^{n} \frac{j!}{(j-i)!} a_j X^{j-i} \quad \text{and} \quad f^{(i)}(X) = \sum_{j=i}^{n} \binom{j}{i} a_j X^{j-i},$$

respectively. Note that $f^{[i]}(X) = i! f^{(i)}(X)$ for all i and f.

Definition 4.13 (Univariate multiplicity codes, from [RT97, Nie01, KSY14])**.** *Let $\mathbb{F}_q$ be a finite field and let s be a positive integer. Let $\alpha_1, \ldots, \alpha_n$ be distinct points in $\mathbb{F}_q$, and let $k < sn$ be a positive integer. The univariate multiplicity code $\mathsf{Mult}_{s,\mathbb{F}_q}(\alpha_1, \ldots, \alpha_n; k)$ is the code over the alphabet $\Sigma = \mathbb{F}_q^s$ of length n which associates each polynomial $f(X) \in (\mathbb{F}_q)_{<k}[X]$ to the codeword $c \in \Sigma^n$ such that for $i \in [n]$,*

$$c_i = (f^{(0)}(\alpha_i), f^{(1)}(\alpha_i), \ldots, f^{(s-1)}(\alpha_i)),$$

where $f^{(j)}$ is the j'th **Hasse derivative** *of f. Let $\mathsf{Mult}_{s,\mathbb{F}_q,k} := \mathsf{Mult}_{s,\mathbb{F}_q}(1, \ldots, q; k)$; note that $\mathsf{Mult}_{s,\mathbb{F}_q,k}$ is $\mathbb{F}_q$-linear.*

Definition 4.14 (List recoverable)**.** *A code $C \subseteq \Sigma^n$ is (ℓ, L)-*list recoverable *if for all $S_1, \ldots, S_n \subseteq \Sigma$ such that $\frac{1}{n}\sum_{i=1}^{n} |S_i| \leq \ell$,*

$$|\{(x_1, \ldots, x_n) \in C : \forall i \in [n], x_i \in S_i\}| \leq L.$$

Definition 4.15 (Expanders [GUV09])**.** *A bipartite graph with N left-vertices, M right-vertices, and left-degree D is specified by a function $\Gamma : [N] \times [D] \to [M]$, where $\Gamma(x, y)$ denotes the y'th neighbor of x. For a set $X \subseteq [N]$, we write $\Gamma(X)$ to denote its set of neighbors $\bigcup_{x \in X, y \in [D]} \Gamma(x, y)$. For a set $T \subseteq [M]$, we write $\mathrm{LIST}_\Gamma(T) = \{x : \Gamma(x) \subseteq T\}$.*

*We say that Γ is a (K, A)-*expander *if for every set $X \subseteq [N]$ of size at most K, $|\Gamma(X)| \geq A \cdot |X|$. Note that if Γ is a (K, A)-expander then for all $B \leq K$ and all sets T such that $|T| < AB$, $|\mathrm{LIST}_\Gamma(T)| < B$.*

Our separation will utilize expanders based on multiplicity codes as constructed in [KTS22].[3]

Theorem 4.16 ([KTS22])**.** *For every field $\mathbb{F}_q$, k, $s \in \mathbb{N}$ such that $15 \leq s + 1 \leq k \leq \mathsf{char}(\mathbb{F}_q)$, identify the elements of $\mathbb{F}_q^k$ with univariate polynomials of degree less than k. Define the graph $\Gamma : \mathbb{F}_q^k \times \mathbb{F}_q \to \mathbb{F}_q^{s+1}$ by*

$$\Gamma(f, y) = (y, f^{[0]}(y), f^{[1]}(y), \ldots, f^{[s-1]}(y)),$$

where $f^{[i]}$ is the i'th **formal derivative** *of f in $\mathbb{F}_q[X]$. For every $K > 0$, Γ is a (K, A)-expander where*

$$A = q - \frac{k(s+1)}{2} \cdot (qK)^{\frac{1}{s+1}}.$$

Corollary 4.17. *For each security parameter $\lambda \in \mathbb{N}$, let $k = \lambda^3$, $\lambda^5 < q \leq 2\lambda^5$ be any prime, and $s = \lambda$. Identify the elements of $\mathbb{F}_q^k$ with univariate polynomials of degree less than k. Define the code $C'_\lambda \subseteq \Sigma^q = (\mathbb{F}_q^s)^q$ with encoding map*

$$\mathtt{Enc}(f) = \{(f^{[0]}(y), f^{[1]}(y), \ldots, f^{[s-1]}(y))\}_{y \in \mathbb{F}_q},$$

where $f^{[i]}$ is the i'th **formal derivative** *of f. Then, for sufficiently large λ, C'_λ is $(\ell, 2\ell)$-list recoverable if $\ell \leq 2^s = 2^\lambda$.*

Proof. Set $K = 2^{s+1}$ and fix $\ell \leq 2^s$; Theorem 4.16 implies that $\Gamma : \mathbb{F}_q^k \times \mathbb{F}_q \to \mathbb{F}_q^{s+1}$ is a (K, A)-expander where

$$A \geq q - \frac{k(s+1)}{2} \cdot (qK)^{\frac{1}{s+1}} = q - \frac{k(s+1)}{2} \cdot (q \cdot 2^{s+1})^{\frac{1}{s+1}} > q/2,$$

for sufficiently large λ.

[3] We note that the proof of expansion extends straightforwardly to subgraphs of Γ defined by taking edges corresponding to subsets $S \subseteq \mathbb{F}_q$, although it suffices for us to take $S = \mathbb{F}_q$.

Fix any lists $S_1, \dots, S_q \subseteq \Sigma$ such that $\frac{1}{q}\sum_{i=1}^{q} |S_i| \le \ell$ and consider the set

$$R = \bigcup_{i=1}^{q} R_i, \quad R_i := \{(i, w_0, \dots, w_{s-1}) : (w_0, \dots, w_{s-1}) \in S_i\}.$$

Observe that R is a set of right vertices of Γ and that $|R| = \sum_i |S_i| \le q\ell < A \cdot 2\ell \le AK$. Now consider the set X of all polynomials f such that for all $i \in [q], \mathtt{Enc}(f)_i \in S_i$; our goal is to bound $|X|$. By construction, $X = \mathrm{LIST}_\Gamma(R)$, so it follows that $|X| = |\mathrm{LIST}_\Gamma(R)| < 2\ell$. □

In our parameter regime, it is straightforward to see that the code $C_\lambda = \mathsf{Mult}_{s,\mathbb{F}_q,k}$ has the same list-recoverability as C'_λ, since for fields $\mathbb{F}_q$ where $\mathsf{char}(\mathbb{F}_q) > s$, C_λ and C'_λ are identical up to scalar factors.

Corollary 4.18. *For each security parameter $\lambda \in \mathbb{N}$, let $k = \lambda^3$, $\lambda^5 < q \le 2\lambda^5$ be any prime, and $s = \lambda$. Then, for sufficiently large λ, $C_\lambda = \mathsf{Mult}_{s,\mathbb{F}_q,k}$ is $(\ell, 2\ell)$-list recoverable for all $\ell \le 2^\lambda$.*

Finally, we will use a result about duals of univariate multiplicity codes which we reprove in Section §A.[4]

Theorem 4.19 ([RZVW24])**.** *For all parameters s, q, and $k < sq$, $(\mathsf{Mult}_{s,\mathbb{F}_q,k})^\perp$ has distance at least $\frac{k+1}{s}$.*

4.4 Yao's Box Problem and Non-Uniform Advice

We will need the following results on non-uniform advice for our separation between $\mathsf{BQP}/\mathrm{qpoly}$ and $\mathsf{BQP}/\mathrm{poly}$.

Theorem 4.20 ([CGLQ20])**.** *Let $G : [N] \to \{0,1\}$ be a random function. Let $\mathcal{A}$ be an unbounded-time algorithm, with S bits of classical advice z_G. For an index $x \in [N]$, let $G|^x : N \to \{0,1\}$ denote the function that results from removing x from G; in other words, on inputs $x' \neq x$, $G(x') = G|^x(x')$ and $G|^x(x) = 0$. The probability that $\mathcal{A}$ computes $G(x)$ while making Q quantum queries to $G|^x$ for a random index x is at most*

$$\Pr_{G,x}[\mathcal{A}^{G|^x}(z_G, x) = G(x)] \le \frac{1}{2} + O\left(\frac{(S + \log N)Q}{N}\right)^{1/3}.$$

Lemma 4.21. *Let $G : \{0,1\}^\lambda \to \{0,1\}$ be a uniformly random function. For an algorithm $\mathcal{A}$ that makes $Q(\lambda) = \mathrm{poly}(\lambda)$ quantum queries to G and a family of $t(\lambda) = \mathrm{poly}(\lambda)$-bit classical advice $\{z_G\}_G$, suppose that*

$$\Pr_{G, x \leftarrow \{0,1\}^n}[\mathcal{A}^G(z_G, x) = G(x)] > \frac{3}{5}.$$

Then, for sufficiently large λ, for a $\frac{1}{4000Q^2}$ fraction of $x \in \{0,1\}^\lambda$, measuring a random query of $\mathcal{A}^G$ (for randomly sampled G) will produce x with probability at least $\frac{1}{3200Q^2}$.

Proof. The proof closely follows [LLPY23], but considers quantum queries instead of classical ones. The only way for $\mathcal{A}$ to distinguish G from $G|^x$ is to have nontrivial query mass at x. Denote by $M_{G,x}$ the total query mass that x is placed by $\mathcal{A}$ when querying G. For each G and x we have that

$$|\Pr[\mathcal{A}^G(z_G, x) = G(x)] - \Pr[\mathcal{A}^{G|^x}(z_G, x) = G(x)]| \le 4\sqrt{QM_{G,x}}.$$

Now we consider the case when we uniform randomly choose $x \leftarrow \{0,1\}^\lambda$, and require $\mathcal{A}^{G|^x}(z_G, x)$ to output $G(x)$. This is exactly Yao's box problem, so by Theorem 4.20,

$$\Pr_{G,x}[\mathcal{A}^{G|^x}(z_G, x) = G(x)] \le \frac{1}{2} + O\left(\frac{(t+\lambda)Q}{2^\lambda}\right)^{1/3} = \frac{1}{2} + \mathrm{negl}(\lambda)$$

$$\implies \Pr_{G,x}[\mathcal{A}^G(z_G, x) = G(x)] - \Pr_{G,x}[\mathcal{A}^{G|^x}(z_G, x) = G(x)] \ge \frac{1}{10} - \mathrm{negl}(\lambda).$$

[4] We note that the work we cite ([RZVW24]) was recently retracted, but the particular theorem we use remains correct. For completeness, an entirely self-contained proof of this fact is given in the appendix.

Thus,

$$\mathop{\mathbb{E}}_{G,x}[M_{G,x}] \geq \mathop{\mathbb{E}}_{G,x}\left[\frac{1}{16Q}\left(\Pr[\mathcal{A}^G(z_G,x)=G(x)]-\Pr[\mathcal{A}^{G^{|x}}(z_G,x)=G(x)]\right)^2\right]$$
$$\geq \frac{1}{16Q}\left(\mathop{\mathbb{E}}_{G,x}[\Pr[\mathcal{A}^G(z_G,x)=G(x)]-\Pr[\mathcal{A}^{G^{|x}}(z_G,x)=G(x)]]\right)^2 \geq \frac{1}{1600Q}-\mathrm{negl}(\lambda),$$

by Jensen's inequality. Finally, by a Markov inequality, we see that

$$\Pr_x\left[\mathop{\mathbb{E}}_G[M_{G,x}] \geq \frac{1}{3200Q}\right] \geq \frac{1}{3200Q^2}-\mathrm{negl}(\lambda).$$

Thus, for sufficiently large λ, for a $\frac{1}{3200Q^2}-\mathrm{negl}(\lambda) \geq \frac{1}{4000Q^2}$ fraction of $x \in \{0,1\}^\lambda$, measuring a random query of $\mathcal{A}^G$ (for randomly sampled G) will produce x with probability at least $\frac{1}{3200Q^2}$. □

5 The Generalized Code Intersection Problem

5.1 Definitions and Basic Facts

We begin by recalling the definitions and basic results from [YZ24]. Much of this section will be taken directly from [YZ24], with only minor modifications. We first define the code intersection relation, which is essentially the problem of finding codewords over n symbols whose symbols have a particular hash value.

Definition 5.1 (Code intersection relation, adapted from [YZ24, LLPY23, BDK24])**.** *For a function $H : [n]\times\Sigma \to \{0,1\}$ and a code $C \subseteq \Sigma^n$, define the code intersection relation $R_{C,H} \subseteq \{0,1\}^n \times \Sigma^n$ by*

$$R_{C,H} = \{(\mathbf{x},\mathbf{v}) = (x_1,\ldots,x_n,v_1,\ldots,v_n) : ((v_1,\ldots,v_n)\in C) \wedge (\forall i \in [n], H(i,v_i)=x_i)\}.$$

Remark 5.2. *We can view $H : [n]\times\Sigma \to \{0,1\}$ as a collection of n many functions, $H(i,\cdot):\Sigma\to\{0,1\}$, and we will at times use the notation $H_i : \Sigma \to \{0,1\}$ when referring to the function corresponding to the i'th output coordinate of H.*

Definition 5.3 (Trace over a finite field [YZ24])**.** *For any prime power $q = r^m$ where r is prime, we define the trace function $\mathrm{Tr}(x) := \sum_{i=0}^{m-1} x^{r^i}$ which maps elements of $\mathbb{F}_q$ to $\mathbb{F}_r$. The trace function is $\mathbb{F}_r$-linear: for all $a,b \in \mathbb{F}_r$ and $x,y \in \mathbb{F}_q$, $\mathrm{Tr}(ax+by) = a\,\mathrm{Tr}(x) + b\,\mathrm{Tr}(y)$. In addition, for any $x \in \mathbb{F}_q^n$, $\sum_{y\in\mathbb{F}_q^n}\omega_r^{\mathrm{Tr}(x\cdot y)} = 0$, where $\omega_r := e^{2\pi i/r}$.*

Definition 5.4 (Quantum Fourier transform over a finite field [YZ24])**.** *For a finite field $\mathbb{F}_q$ where $q = r^m$ and r is prime, the quantum Fourier transform over $\mathbb{F}_q$ is the unitary denoted by QFT_q such that for any $x \in \mathbb{F}_q$,*

$$\mathrm{QFT}_q|x\rangle = \frac{1}{\sqrt{q}}\sum_{z\in\mathbb{F}_q}\omega_r^{\mathrm{Tr}(x\cdot z)}|z\rangle.$$

The QFT over an alphabet $\Sigma = \mathbb{F}_q^s$ is the s-wise tensor product of QFT_q: for $\mathbf{x} = (x_1,\ldots,x_s) \in \Sigma$,

$$\mathrm{QFT}_\Sigma|\mathbf{x}\rangle := \mathrm{QFT}_q^{\otimes s}|x_1\rangle\ldots|x_s\rangle = \frac{1}{\sqrt{|\Sigma|}}\sum_{\mathbf{z}\in\Sigma}\omega_r^{\mathrm{Tr}(\mathbf{x}\cdot\mathbf{z})}|\mathbf{z}\rangle.$$

Similarly, for any positive integer n and $\mathbf{x} \in \Sigma^n$, we have

$$\mathrm{QFT}_\Sigma^{\otimes n}|\mathbf{x}\rangle = \frac{1}{|\Sigma|^{n/2}}\sum_{\mathbf{z}\in\Sigma^n}\omega_r^{\mathrm{Tr}(\mathbf{x}\cdot\mathbf{z})}|\mathbf{z}\rangle.$$

The unitary QFT_q can be approximated within error ε in operator norm in time $\mathrm{poly}(\log q, \log 1/\varepsilon)$ [CW02, vDHI06].

Definition 5.5 (Fourier transform of a function [YZ24])**.** *For functions $f, g : \Sigma^n \to \mathbb{C}$, we define*

$$\hat{f}(\mathbf{z}) := \frac{1}{|\Sigma|^{n/2}} \sum_{\mathbf{x}\in\Sigma^n} f(\mathbf{x})\omega_r^{\mathrm{Tr}(\mathbf{x}\cdot\mathbf{z})}, \quad (f \cdot g)(\mathbf{x}) := f(\mathbf{x}) \cdot g(\mathbf{x}), \text{ and } (f \star g)(\mathbf{x}) := \sum_{\mathbf{y}\in\Sigma^n} f(\mathbf{y}) \cdot g(\mathbf{x}-\mathbf{y}).$$

Note that

$$\mathrm{QFT}_\Sigma^{\otimes n} \sum_{\mathbf{x}\in\Sigma^n} f(\mathbf{x}) |\mathbf{x}\rangle = \sum_{\mathbf{z}\in\Sigma^n} \hat{f}(\mathbf{z}) |\mathbf{z}\rangle .$$

Fact 5.6 ([YZ24])**.** *The following properties hold for the Fourier transform:*

1. *(Parseval's equality) For all functions $f : \Sigma^n \to \mathbb{C}$, $\sum_{\mathbf{x}\in\Sigma^n} |f(\mathbf{x})|^2 = \sum_{\mathbf{z}\in\Sigma^n} |\hat{f}(\mathbf{z})|^2$.*
2. *(Pointwise transform) Suppose that we have $f_i : \Sigma \to \mathbb{C}$ for $i \in [n]$ and $f : \Sigma^n \to \mathbb{C}$ is defined by $f(\mathbf{x}) := \prod_{i=1}^n f_i(x_i)$. Then $\hat{f}(\mathbf{z}) = \prod_{i=1}^n \hat{f}_i(z_i)$.*
3. *(Convolution theorem) For all functions $f, g, h : \Sigma^n \to \mathbb{C}$, $\widehat{f \cdot g} = \frac{1}{|\Sigma|^{n/2}}(\hat{f} \star \hat{g})$, $\widehat{f \star g} = |\Sigma|^{n/2}(\hat{f} \cdot \hat{g})$, and $f \cdot \widehat{(g \star h)} = (\hat{f} \star (\hat{g} \cdot \hat{h}))$.*

Lemma 5.7 (Fourier transform of a linear code)**.** *Let $C \subseteq \Sigma^n = (\mathbb{F}_q^s)^n$ be any $\mathbb{F}_q$-linear code. Then,*

$$f(\mathbf{u}) = \begin{cases} \frac{1}{\sqrt{|C|}} & \text{if } \mathbf{u} \in C, \\ 0 & \text{otherwise.} \end{cases} \iff \hat{f}(\mathbf{u}) = \begin{cases} \frac{1}{\sqrt{|C^\perp|}} & \text{if } \mathbf{u} \in C^\perp, \\ 0 & \text{otherwise.} \end{cases}$$

Proof. Since C is $\mathbb{F}_q$-linear, $|C| \cdot |C^\perp| = |\Sigma|^n$, and thus for any $\mathbf{z} \in C^\perp$,

$$\hat{f}(\mathbf{z}) = \frac{1}{|\Sigma|^{n/2}} \sum_{\mathbf{u}\in\Sigma^n} f(\mathbf{u})\omega_r^{\mathrm{Tr}(\mathbf{u}\cdot\mathbf{z})} = \frac{1}{|\Sigma|^{n/2}} \sum_{\mathbf{u}\in C} \frac{1}{\sqrt{|C|}} = \frac{1}{\sqrt{|C^\perp|}}.$$

Finally, $\hat{f}(\mathbf{z}) = 0$ for $\mathbf{z} \notin C^\perp$ by Parseval's equality. The reverse direction follows by an identical argument. □

Lemma 5.8. *Let $|\psi\rangle, |\phi\rangle$ be states such that $\||\psi\rangle\| = 1$ and $\||\psi\rangle - |\phi\rangle\| \le \varepsilon < 1$. Then, $\left\||\psi\rangle - \frac{|\phi\rangle}{\||\phi\rangle\|}\right\| \le 2\varepsilon$.*

Proof. By the reverse/inverse triangle inequality,

$$|\| |\phi\rangle \| - 1| = |\| |\phi\rangle \| - \| |\psi\rangle \|| \le \| |\phi\rangle - |\psi\rangle \| = \varepsilon \implies \| |\phi\rangle \| \ge 1 - \varepsilon > 0.$$

We can thus define the normalized state $|\phi'\rangle := \frac{|\phi\rangle}{\||\phi\rangle\|}$. By the regular triangle inequality,

$$\| |\psi\rangle - |\phi'\rangle \| \le \| |\psi\rangle - |\phi\rangle \| + \| |\phi\rangle - |\phi'\rangle \| \le \varepsilon + |\| |\phi\rangle \| - 1| \le 2\varepsilon.$$ □

Lemma 5.9 ([BV93])**.** *Let $|\psi\rangle, |\phi\rangle$ be states such that $\| |\psi\rangle \| = \| |\phi\rangle \| = 1$ and $\| |\psi\rangle - |\phi\rangle \| \le \varepsilon$. Then the total variation distance between the probability distributions resulting from measurements of $|\phi\rangle$ and $|\psi\rangle$ is at most 4ε.*

Now we state the main algorithmic result of [YZ24], namely that a quantum algorithm can approximately implement the convolution trick for some families of functions.

Lemma 5.10 ([YZ24])**.** *Let $|\psi\rangle$ and $|\phi\rangle$ be quantum states on a quantum system over an alphabet $\Sigma = \mathbb{F}_q^s$ written as*

$$|\psi\rangle = \sum_{\mathbf{u}\in\Sigma^n} V(\mathbf{u}) |\mathbf{u}\rangle \quad \text{and} \quad |\phi\rangle = \sum_{\mathbf{e}\in\Sigma^n} W(\mathbf{e}) |\mathbf{e}\rangle ,$$

for functions $V, W : \Sigma^n \to \mathbb{C}$. Let $F : \Sigma^n \to \Sigma^n$ be a function and let $\mathsf{GOOD} \subseteq \Sigma^n \times \Sigma^n$ be a subset such that for any $(\mathbf{u}, \mathbf{e}) \in \mathsf{GOOD}$, we have $F(\mathbf{u} + \mathbf{e}) = \mathbf{u}$. Define $\mathsf{BAD} := (\Sigma^n \times \Sigma^n) \setminus \mathsf{GOOD}$. Suppose that

$$\sum_{(\mathbf{u},\mathbf{e})\in\mathsf{BAD}} |\widehat{V}(\mathbf{u})\widehat{W}(\mathbf{e})|^2 \le \varepsilon \quad \text{and} \quad \sum_{\mathbf{z}\in\Sigma^n} \left| \sum_{(\mathbf{u},\mathbf{e})\in\mathsf{BAD}:\mathbf{u}+\mathbf{e}=\mathbf{z}} \widehat{V}(\mathbf{u})\widehat{W}(\mathbf{e}) \right|^2 \le \delta .$$

Define the unitaries U_{add} and U_F as follows:

$$U_{\mathsf{add}} = \sum_{(\mathbf{u},\mathbf{e})\in\Sigma^n\times\Sigma^n} |\mathbf{u}+\mathbf{e},\mathbf{u}\rangle\langle\mathbf{e},\mathbf{u}| \quad \textit{and} \quad U_F = \sum_{(\mathbf{u},\mathbf{w})\in\Sigma^n\times\Sigma^n} |\mathbf{w},\mathbf{u}-F(\mathbf{w})\rangle\langle\mathbf{w},\mathbf{u}| \ .$$

Then,

$$\left\| \left(\mathrm{QFT}_\Sigma^{-1,\otimes n}\otimes \mathrm{id}\right)\cdot U_F\cdot U_{\mathsf{add}}\cdot \left(\mathrm{QFT}_\Sigma^{\otimes n}\otimes \mathrm{QFT}_\Sigma^{\otimes n}\right)|\phi\rangle\,|\psi\rangle - |\Sigma|^{n/2}\sum_{\mathbf{z}\in\Sigma^n}(V(\mathbf{z})\cdot W(\mathbf{z}))\,|\mathbf{z}\rangle\,|0\rangle \right\| \le \sqrt{\varepsilon}+\sqrt{\delta}\,.$$

Lemmas 5.8 to 5.10 imply the following corollary (as the QFT, U_{add}, and U_F are unitaries).

Corollary 5.11. *Let V and W be functions, and ϵ and δ be the corresponding error parameters from Lemma 5.10. For any property $\mathcal{P}:\Sigma^n\to\{0,1\}$, if $\|\,|\phi\rangle\,\| = \|\,|\psi\rangle\,\| = 1$ and measuring (in the standard basis) the normalization of*

$$|\Sigma|^{n/2}\sum_{\mathbf{z}\in\Sigma^n}(V(\mathbf{z})\cdot W(\mathbf{z}))\,|\mathbf{z}\rangle\,|0\rangle \ ,$$

produces an outcome $\mathbf{z}$ such that $\mathcal{P}(\mathbf{z})=1$ with probability p, then measuring (in the standard basis)

$$\left(\mathrm{QFT}_\Sigma^{-1,\otimes n}\otimes \mathrm{id}\right)\cdot U_F\cdot U_{\mathsf{add}}\cdot \left(\mathrm{QFT}_\Sigma^{\otimes n}\otimes \mathrm{QFT}_\Sigma^{\otimes n}\right)|\phi\rangle\,|\psi\rangle \ ,$$

produces an outcome $\mathbf{z}$ such that $\mathcal{P}(\mathbf{z})=1$ with probability at least $p-8(\sqrt{\varepsilon}+\sqrt{\delta})$.

5.2 Technical Lemmas

In this section, we state some technical lemmas to extend the Yamakawa-Zhandry algorithm to work with biased oracles. We first introduce a pair of functions V and W that represent normalized indicators for a code C and the preimages of any output $\mathbf{b}$ for a function H. For any $\mathbb{F}_q$-linear code $C\subseteq\Sigma^n=(\mathbb{F}_q^s)^n$, function $H:[n]\times\Sigma\to\{0,1\}$, and string $\mathbf{b}\in\{0,1\}^n$, let $V:\Sigma^n\to\mathbb{C}$, $W_i^{H_i,b_i}:\Sigma\to\mathbb{C}$, and $W^{H,\mathbf{b}}:\Sigma^n\to\mathbb{C}$ be defined as follows:

$$V(\mathbf{u}) = \begin{cases} \frac{1}{\sqrt{|C|}} & \mathbf{u}\in C \\ 0 & \text{otherwise}\end{cases},$$

$$W_i^{H_i,b_i}(e) = \begin{cases} \frac{1}{\sqrt{|T_i^{H_i,b_i}|}} & e\in T_i^{H_i,b_i} \\ 0 & \text{otherwise}\end{cases},$$

$$W^{H,\mathbf{b}}(e_1,\ldots,e_n) = \prod_{i=1}^n W_i^{H_i,b_i}(e_i)\,,$$

where $T_i^{H_i,b_i}\subseteq\Sigma$ is the subset consisting of $e_i\in\Sigma$ such that $H_i(e_i)=b_i$.

Definition 5.12 (p-biased distribution)**.** *For any $p\in[0,1]$ and set Σ, let $\mathsf{Bias}_{p,\Sigma}$ denote the distribution over functions from Σ to $\{0,1\}$ that samples $F:\Sigma\to\{0,1\}$ with probability $p^{|F^{-1}(1)|}(1-p)^{|F^{-1}(0)|}$. Let $\mathsf{Bias}_{n,p,\Sigma}$ denote the distribution over functions $G:[n]\times\Sigma\to\{0,1\}$ that samples G with probability $\mathsf{Bias}_{n,p,\Sigma}(G) := \prod_{i=1}^n \mathsf{Bias}_{p,\Sigma}(G_i)$.*

The following claim follows immediately from the definition of $\mathsf{Bias}_{n,p,\Sigma}$.

Claim 5.13. *Let π be any permutation over Σ (resp. Σ^n). Then, the distributions $\mathsf{Bias}_{p,\Sigma}$ and $\mathsf{Bias}_{p,\Sigma}\circ\pi$ (resp. $\mathsf{Bias}_{n,p,\Sigma}$ and $\mathsf{Bias}_{n,p,\Sigma}\circ\pi$) are identical.*

The following lemma shows that when we take the Fourier transform of the preimage state of H_i sampled from $\mathsf{Bias}_{p,\Sigma}$, the resulting Fourier coefficients are uniform over all non-zero elements of Σ, and have constant weight (either p or $1-p$ depending on if we are taking the preimage of 0 or 1) on 0.

Claim 5.14. *Fix any string $\mathbf{b} \in \{0,1\}^n$, for all $i \in [n]$ and $\sigma, \sigma' \in \Sigma \setminus \{0\}$, it holds that*

$$\mathop{\mathbb{E}}_{H_i \leftarrow \mathsf{Bias}_{p,\Sigma}}[|\widehat{W}_i^{H_i,b_i}(0)|^2] = \begin{cases} p & \text{if } b_i = 1 \\ 1-p & \text{if } b_i = 0 \end{cases} \quad \text{and} \quad \mathop{\mathbb{E}}_{H_i \leftarrow \mathsf{Bias}_{p,\Sigma}}[|\widehat{W}_i^{H_i,b_i}(\sigma)|^2] = \mathop{\mathbb{E}}_{H_i \leftarrow \mathsf{Bias}_{p,\Sigma}}[|\widehat{W}_i^{H_i,b_i}(\sigma')|^2].$$

Proof. We can directly compute the expected Fourier weight on 0 as follows:

$$\mathop{\mathbb{E}}_{H_i \leftarrow \mathsf{Bias}_{p,\Sigma}}\left[\left|\widehat{W}_i^{H_i,b_i}(0)\right|^2\right] = \mathop{\mathbb{E}}_{H_i \leftarrow \mathsf{Bias}_{p,\Sigma}}\left[\left|\frac{1}{\sqrt{|\Sigma|}}\sum_{z\in\Sigma} W_i^{H_i,b_i}(z)\right|^2\right] = \frac{\mathbb{E}_{H_i}\left[\left|T_i^{H_i,b_i}\right|\right]}{|\Sigma|} = \begin{cases} p & \text{if } b_i = 1 \\ 1-p & \text{if } b_i = 0 \end{cases}.$$

Since $\sigma \neq 0$ (resp. $\sigma' \neq 0$), for any $w \in \mathbb{F}_q$, the number of $z \in \Sigma$ such that $\sigma \cdot z = w$ is $|\Sigma|/q$. Therefore, there is a permutation $\pi_{e,e'} : \Sigma \to \Sigma$ such that $\sigma \cdot z = \sigma' \cdot \pi_{\sigma,\sigma'}(z)$ for all $z \in \Sigma$. Thus, by Claim 5.13,

$$\begin{aligned}
\mathop{\mathbb{E}}_{H_i \leftarrow \mathsf{Bias}_{p,\Sigma}}\left[\left|\widehat{W}_i^{H_i,b_i}(\sigma)\right|^2\right] &= \mathop{\mathbb{E}}_{H_i \leftarrow \mathsf{Bias}_{p,\Sigma}}\left[\left|\frac{1}{\sqrt{|\Sigma|}}\sum_{z\in\Sigma} W_i^{H_i,b_i}(z)\cdot\omega_r^{\mathrm{Tr}(\sigma\cdot z)}\right|^2\right] \\
&= \mathop{\mathbb{E}}_{H_i \leftarrow \mathsf{Bias}_{p,\Sigma}}\left[\left|\frac{1}{\sqrt{|\Sigma|}}\sum_{z\in\Sigma} W_i^{H_i\circ\pi^{-1}_{\sigma,\sigma'},b_i}(\pi_{\sigma,\sigma'}(z))\cdot\omega_r^{\mathrm{Tr}\left(\sigma'\cdot\pi_{\sigma,\sigma'}(z)\right)}\right|^2\right] \\
&= \mathop{\mathbb{E}}_{H_i \leftarrow \mathsf{Bias}_{p,\Sigma}}\left[\left|\frac{1}{\sqrt{|\Sigma|}}\sum_{z\in\Sigma} W_i^{H_i\circ\pi^{-1}_{\sigma,\sigma'},b_i}(z)\cdot\omega_r^{\mathrm{Tr}(\sigma'\cdot z)}\right|^2\right] \\
&= \mathop{\mathbb{E}}_{H_i \leftarrow \mathsf{Bias}_{p,\Sigma}}\left[\left|\frac{1}{\sqrt{|\Sigma|}}\sum_{z\in\Sigma} W_i^{H_i,b_i}(z)\cdot\omega_r^{\mathrm{Tr}(\sigma'\cdot z)}\right|^2\right] \\
&= \mathop{\mathbb{E}}_{H_i \leftarrow \mathsf{Bias}_{p,\Sigma}}\left[\left|\widehat{W}_i^{H_i,b_i}(\sigma')\right|^2\right].
\end{aligned}$$

□

For any function $\mathtt{Dec}_{C^\perp} : \Sigma^n \to \Sigma^n$, define the sets $\mathcal{G} := \{\mathbf{e} \in \Sigma^n : \forall \mathbf{u} \in C^\perp, \mathtt{Dec}_{C^\perp}(\mathbf{u}+\mathbf{e}) = \mathbf{u}\}$, $\mathcal{B} := \Sigma^n \setminus \mathcal{G}$, $\mathsf{GOOD} := C^\perp \times \mathcal{G}$ and $\mathsf{BAD} := (\Sigma^n \times \Sigma^n) \setminus \mathsf{GOOD}$. By construction, $\mathtt{Dec}_{C^\perp}(\mathbf{u}+\mathbf{e}) = \mathbf{u}$ for all $(\mathbf{u},\mathbf{e}) \in \mathsf{GOOD}$. Applying the definition of $\widehat{V}$ from Lemma 5.7, we see that for all $\mathbf{b} \in \{0,1\}^n$ and functions H,

$$\sum_{(\mathbf{u},\mathbf{e})\in\mathsf{BAD}}\left|\widehat{V}(\mathbf{u})\widehat{W}^{H,\mathbf{b}}(\mathbf{e})\right|^2 = \sum_{\mathbf{u}\in C^\perp}\sum_{\mathbf{e}\in\mathcal{B}}\left|\frac{1}{\sqrt{|C^\perp|}}\widehat{W}^{H,\mathbf{b}}(\mathbf{e})\right|^2 = \sum_{\mathbf{e}\in\mathcal{B}}\left|\widehat{W}^{H,\mathbf{b}}(\mathbf{e})\right|^2.$$

We can apply the same logic to get that

$$\sum_{\mathbf{z}\in\Sigma^n}\left|\sum_{(\mathbf{u},\mathbf{e})\in\mathsf{BAD}:\mathbf{u}+\mathbf{e}=\mathbf{z}}\widehat{V}(\mathbf{u})\cdot\widehat{W}^{H,\mathbf{b}}(\mathbf{e})\right|^2 = \sum_{\mathbf{z}\in\Sigma^n}\left|\sum_{\substack{\mathbf{u}\in C^\perp,\mathbf{e}\in\mathcal{B}:\\ \mathbf{u}+\mathbf{e}=\mathbf{z}}}\widehat{V}(\mathbf{u})\cdot\widehat{W}^{H,\mathbf{b}}(\mathbf{e})\right|^2.$$

Definition 5.15. *Let $\mathcal{D}_{p,b}$ be the distribution over Σ that takes 0 with probability $1-p$ if $b = 0$ (resp. probability p if $b = 1$) and otherwise takes a uniformly random element of $\Sigma \setminus \{0\}$. For any bitstring $\mathbf{b} \in \{0,1\}^n$, define the distribution $\mathcal{D}_{p,\mathbf{b}}$ over Σ^n to be the Cartesian product of the distributions $\mathcal{D}_{p,\mathbf{b}_i}$.*

Definition 5.16. *Fix any $\mathbb{F}_q$-linear code $C \subseteq \Sigma^n$, function $\mathtt{Dec}_{C^\perp} : \Sigma^n \to \Sigma^n$, set $S \subseteq \{0,1\}^n$, $p \in [0,1]$, and real-valued function $\mu : \mathbb{N} \to \mathbb{R}$. $(C, \mathtt{Dec}_{C^\perp})$ is said to be (p,μ,S)-good if for all $\lambda \in \mathbb{N}$ and $\mathbf{b} \in S$, $\Pr_{\mathbf{e}\leftarrow\mathcal{D}_{p,\mathbf{b}}}[\mathbf{e} \in \mathcal{B}] \leq \mu(\lambda)$.*

Lemma 5.17. *Suppose that $(C, \mathtt{Dec}_{C^\perp})$ is (p,μ,S)-good. Then, for all $\lambda \in \mathbb{N}$ and $\mathbf{b} \in S$,*

$$\mathop{\mathbb{E}}_{H\leftarrow\mathsf{Bias}_{n,p,\Sigma}}\left[\sum_{\mathbf{e}\in\mathcal{B}}\left|\widehat{W}^{H,\mathbf{b}}(\mathbf{e})\right|^2\right] \leq \mu(\lambda).$$

Proof. By Claim 5.14, $\mathcal{D}_{p,b_i}(e_i) = \mathbb{E}_{H_i}[|\widehat{W}_i^{H_i,b_i}(e_i)|^2]$ for all $e_i \in \Sigma$ where (slightly abusing notation) $\mathcal{D}_{p,b_i}(\cdot)$ is the density function of the distribution $\mathcal{D}_{p,b_i}$. Moreover, for any $\mathbf{e} = (e_1, \ldots, e_n) \in \Sigma^n$, string $\mathbf{b}$ and function H, since $W^{H,\mathbf{b}}(\mathbf{e}) = \prod_{i=1}^n W_i^{H_i,b_i}(e_i)$, by Fact 5.6, we have that $\widehat{W}^{H,\mathbf{b}}(\mathbf{e}) = \prod_{i=1}^n \widehat{W}_i^{H_i,b_i}(e_i)$. Thus, $\mathcal{D}_{p,\mathbf{b}}(\mathbf{e}) = \mathbb{E}_{H \leftarrow \mathsf{Bias}_{n,p,\Sigma}}[|\widehat{W}^{H,\mathbf{b}}(\mathbf{e})|^2]$ for all $\mathbf{e} \in \Sigma^n$. By linearity of expectation, we see that for all $\lambda \in \mathbb{N}$ and $\mathbf{b} \in S$,

$$\mathop{\mathbb{E}}_{H \leftarrow \mathsf{Bias}_{n,p,\Sigma}}\left[\sum_{\mathbf{e}\in\mathcal{B}} \left|\widehat{W}^{H,\mathbf{b}}(\mathbf{e})\right|^2\right] = \sum_{\mathbf{e}\in\mathcal{B}} \mathop{\mathbb{E}}_{H \leftarrow \mathsf{Bias}_{n,p,\Sigma}}\left[\left|\widehat{W}^{H,\mathbf{b}}(\mathbf{e})\right|^2\right] = \sum_{\mathbf{e}\in\mathcal{B}} \mathcal{D}_{p,\mathbf{b}}(\mathbf{e}) = \Pr_{\mathbf{e}\leftarrow\mathcal{D}_{p,\mathbf{b}}}[\mathbf{e} \in \mathcal{B}] \le \mu(\lambda)\,. \qquad \square$$

We define the function $B : \Sigma^n \to \mathbb{C}$ to be the inverse Fourier transform of the indicator function $\widehat{B}(\mathbf{e}) = \mathbb{1}_{\mathbf{e}\in\mathcal{B}}$.

Claim 5.18. *Suppose that $(C, \mathsf{Dec}_{C^\perp})$ is (p, μ, S)-good. Then for all $\lambda \in \mathbb{N}$, $\mathbf{b} \in S$, and $\mathbf{z} \in \Sigma^n$,*

$$\mathop{\mathbb{E}}_{H \leftarrow \mathsf{Bias}_{n,p,\Sigma}}\left[\left|(B \star W^{H,\mathbf{b}})(\mathbf{z})\right|^2\right] \le \mu(\lambda).$$

Proof. For $\mathbf{z}_0, \mathbf{z}_1 \in \Sigma^n$, define the permutation $\pi_{\mathbf{z}_0,\mathbf{z}_1} : \Sigma^n \to \Sigma^n$ as $\pi_{\mathbf{z}_0,\mathbf{z}_1}(\mathbf{z}) := \mathbf{z} + \mathbf{z}_0 - \mathbf{z}_1$. By Claim 5.13,

$$\begin{aligned}
\mathop{\mathbb{E}}_{H \leftarrow \mathsf{Bias}_{n,p,\Sigma}}\left[\left|(B \star W^{H,\mathbf{b}})(\mathbf{z}_0)\right|^2\right] &= \mathop{\mathbb{E}}_{H \leftarrow \mathsf{Bias}_{n,p,\Sigma}}\left[\left|\sum_{\mathbf{x}\in\Sigma^n} B(\mathbf{x}) \cdot W^{H,\mathbf{b}}(\mathbf{z}_0 - \mathbf{x})\right|^2\right] \\
&= \mathop{\mathbb{E}}_{H \leftarrow \mathsf{Bias}_{n,p,\Sigma}}\left[\left|\sum_{\mathbf{x}\in\Sigma^n} B(\mathbf{x}) \cdot W^{H\circ\pi_{\mathbf{z}_0,\mathbf{z}_1},\mathbf{b}}(\mathbf{z}_1 - \mathbf{x})\right|^2\right] \\
&= \mathop{\mathbb{E}}_{H \leftarrow \mathsf{Bias}_{n,p,\Sigma}}\left[\left|\sum_{\mathbf{x}\in\Sigma^n} B(\mathbf{x}) \cdot W^{H,\mathbf{b}}(\mathbf{z}_1 - \mathbf{x})\right|^2\right] \\
&= \mathop{\mathbb{E}}_{H \leftarrow \mathsf{Bias}_{n,p,\Sigma}}\left[\left|(B \star W^{H,\mathbf{b}})(\mathbf{z}_1)\right|^2\right].
\end{aligned}$$

Thus, by Lemma 5.17, we have that for all $\lambda \in \mathbb{N}$, $\mathbf{b} \in S$, and $\mathbf{z} \in \Sigma^n$,

$$\begin{aligned}
\mathop{\mathbb{E}}_{H \leftarrow \mathsf{Bias}_{n,p,\Sigma}}\left[\left|(B \star W^{H,\mathbf{b}})(\mathbf{z})\right|^2\right] &= \frac{1}{|\Sigma|^n} \mathop{\mathbb{E}}_{H \leftarrow \mathsf{Bias}_{n,p,\Sigma}}\left[\sum_{\mathbf{z}\in\Sigma^n} \left|(B \star W^{H,\mathbf{b}})(\mathbf{z})\right|^2\right] \\
&= \frac{1}{|\Sigma|^n} \mathop{\mathbb{E}}_{H \leftarrow \mathsf{Bias}_{n,p,\Sigma}}\left[\sum_{\mathbf{z}\in\Sigma^n} \left||\Sigma|^{n/2}\widehat{B}(\mathbf{z}) \cdot \widehat{W}^{H,\mathbf{b}}(\mathbf{z})\right|^2\right] \\
&= \mathop{\mathbb{E}}_{H \leftarrow \mathsf{Bias}_{n,p,\Sigma}}\left[\sum_{\mathbf{z}\in\Sigma^n} \left|\widehat{B}(\mathbf{z}) \cdot \widehat{W}^{H,\mathbf{b}}(\mathbf{z})\right|^2\right] \\
&= \mathop{\mathbb{E}}_{H \leftarrow \mathsf{Bias}_{n,p,\Sigma}}\left[\sum_{\mathbf{z}\in\mathcal{B}} \left|\widehat{W}^{H,\mathbf{b}}(\mathbf{z})\right|^2\right] \le \mu(\lambda)\,.
\end{aligned} \qquad \square$$

Claim 5.19. *For any function H and string $\mathbf{b} \in \{0,1\}^n$, it holds that*

$$\sum_{\mathbf{z}\in\Sigma^n} \left|\sum_{\substack{\mathbf{u}\in C^\perp, \mathbf{e}\in\mathcal{B}: \\ \mathbf{u}+\mathbf{e}=\mathbf{z}}} \widehat{V}(\mathbf{u}) \cdot \widehat{W}^{H,\mathbf{b}}(\mathbf{e})\right|^2 = \sum_{\mathbf{z}\in\Sigma^n} \left|(V \cdot (B \star W^{H,\mathbf{b}}))(\mathbf{z})\right|^2.$$

Proof. For any $\mathbf{z} \in \Sigma^n$, we use the fact that $V(\mathbf{x}) = 0$ for $\mathbf{x} \notin C^\perp$ to show that

$$\sum_{\substack{\mathbf{u}\in C^\perp, \mathbf{e}\in\mathcal{B}: \\ \mathbf{u}+\mathbf{e}=\mathbf{z}}} \widehat{V}(\mathbf{u}) \cdot \widehat{W}^{H,\mathbf{b}}(\mathbf{e}) = \sum_{\substack{\mathbf{u}\in\Sigma^n, \mathbf{e}\in\Sigma^n: \\ \mathbf{u}+\mathbf{e}=\mathbf{z}}} \widehat{V}(\mathbf{u}) \cdot \widehat{B}(\mathbf{e}) \cdot \widehat{W}^{H,\mathbf{b}}(\mathbf{e}) = (\widehat{V} \star (\widehat{B} \cdot \widehat{W}^{H,\mathbf{b}}))(\mathbf{z}) = (V \cdot (\widehat{B \star W^{H,\mathbf{b}}}))(\mathbf{z})\,.$$

The claim then follows from Parseval's equality. $\square$

Corollary 5.20. *Suppose that $(C, \mathtt{Dec}_{C^\perp})$ is (p, μ, S)-good. Then for all $\lambda \in \mathbb{N}$ and $\mathbf{b} \in S$,*

$$\mathop{\mathbb{E}}_{H \leftarrow \mathsf{Bias}_{n,p,\Sigma}} \left[\sum_{\mathbf{z} \in \Sigma^n} \left| \sum_{\substack{\mathbf{u} \in C^\perp, \mathbf{e} \in \mathcal{B}: \\ \mathbf{u}+\mathbf{e}=\mathbf{z}}} \widehat{V}(\mathbf{u}) \cdot \widehat{W}^{H,\mathbf{b}}(\mathbf{e}) \right|^2 \right] \leq \mu(\lambda).$$

Proof. By Claims 5.18 and 5.19, we have that for all $\lambda \in \mathbb{N}$ and $\mathbf{b} \in S$,

$$\begin{aligned}
\mathop{\mathbb{E}}_{H \leftarrow \mathsf{Bias}_{n,p,\Sigma}} \left[\sum_{\mathbf{z} \in \Sigma^n} \left| \sum_{\substack{\mathbf{u} \in C^\perp, \mathbf{e} \in \mathcal{B}: \\ \mathbf{u}+\mathbf{e}=\mathbf{z}}} \widehat{V}(\mathbf{u}) \cdot \widehat{W}^{H,\mathbf{b}}(\mathbf{e}) \right|^2 \right] &= \mathop{\mathbb{E}}_{H \leftarrow \mathsf{Bias}_{n,p,\Sigma}} \left[\sum_{\mathbf{z} \in \Sigma^n} \left| (V \cdot (B \star W^{H,\mathbf{b}}))(\mathbf{z}) \right|^2 \right] \\
&= \mathop{\mathbb{E}}_{H \leftarrow \mathsf{Bias}_{n,p,\Sigma}} \left[\sum_{\mathbf{z} \in C} \frac{1}{|C|} \left| (B \star W^{H,\mathbf{b}})(\mathbf{z}) \right|^2 \right] \\
&= \frac{1}{|C|} \sum_{\mathbf{z} \in C} \mathop{\mathbb{E}}_{H \leftarrow \mathsf{Bias}_{n,p,\Sigma}} \left[\left| (B \star W^{H,\mathbf{b}})(\mathbf{z}) \right|^2 \right] \\
&\leq \frac{1}{|C|} \sum_{\mathbf{z} \in C} \mu(\lambda) = \mu(\lambda)\,.
\end{aligned}$$

□

5.3 The Biased Yamakawa-Zhandry Algorithm

We are now ready to present our modified algorithm for handling biased oracles.

Definition 5.21 (The Yamakawa-Zhandry advice state)**.** *For a code $C \subseteq \Sigma^n$ and function $H : [n] \times \Sigma \to \{0,1\}$, define the sets $S_{i,b} := \{e \in \Sigma : H(i,e) = b\}$ for $(i,b) \in [n] \times \{0,1\}$. Let $|\phi_{i,b}\rangle$ and $|\psi\rangle$ denote the following states:*

$$|\phi_{i,b}\rangle = \begin{cases} \frac{1}{\sqrt{|S_{i,b}|}} \sum_{e \in S_{i,b}} |e\rangle & \text{if } |S_{i,b}| \neq 0 \\ |\perp\rangle & \text{otherwise} \end{cases} \quad \text{and} \quad |\psi\rangle = \frac{1}{\sqrt{|C|}} \sum_{\mathbf{u} \in C} |\mathbf{u}\rangle\,.$$

We define the advice state *for (C, H), denoted by $|\mathsf{adv}_{C,H}\rangle$ (or $|\mathsf{adv}_H\rangle$ when the code C is implicit) as follows:*

$$|\mathsf{adv}_H\rangle := \begin{cases} (\bigotimes_{i=1}^n |\phi_{i,0}\rangle \otimes |\phi_{i,1}\rangle) \otimes |\psi\rangle & \text{if } |\phi_{i,b}\rangle \neq |\perp\rangle \text{ for all } (i,b) \in [n] \times \{0,1\}, \\ |\perp\rangle & \text{otherwise.} \end{cases}$$

Theorem 5.22. *Fix any $\mathbb{F}_q$-linear code $C \subseteq \Sigma^n = (\mathbb{F}_q^s)^n$, function $\mathtt{Dec}_{C^\perp} : \Sigma^n \to \Sigma^n$, set $S \subseteq \{0,1\}^n$, $p \in [0, 1/2]$, and function $\mu : \mathbb{N} \to [0,1]$ such that $(C, \mathtt{Dec}_{C^\perp})$ is (p, μ, S)-good. Let $\mathsf{BiasedYZ} = \mathsf{BiasedYZ}_{\mathtt{Dec}_{C^\perp}}$ be the quantum algorithm described in Figure 1 and $|\mathsf{adv}_H\rangle$ be the state described in Definition 5.21. Then, for all $\lambda \in \mathbb{N}$:*

1. $\mathsf{BiasedYZ}$ *runs in time* $\mathrm{poly}(n, s, \log q, \lambda) + T_{\mathtt{Dec}}$*, where $T_{\mathtt{Dec}}$ is the time required to compute $\mathtt{Dec}_{C^\perp}$.*
2. *For all C and H, $|\mathsf{adv}_H\rangle$ is a $O(sn \log q)$-qubit state.*
3. *For all strings $\mathbf{b} \in S$,*

$$\begin{aligned}
\Pr_{H \leftarrow \mathsf{Bias}_{n,p,\Sigma}} \left[\Pr_{\mathsf{BiasedYZ}} [(\mathbf{b}, \mathbf{v}) \in R_{C,H} : \mathbf{v} \leftarrow \mathsf{BiasedYZ}(|\mathsf{adv}_H\rangle, \mathbf{b})] \geq 1 - 16\mu(\lambda)^{1/4} - 2^{-4\lambda} \right] \\
\geq 1 - (2n(1-p)^{|\Sigma|} + 2\mu(\lambda)^{1/2})\,.
\end{aligned}$$

Proof. Throughout this proof, we assume that $2\mu(\lambda)^{1/4} < 1$, since the theorem holds trivially otherwise. Upon inspection, we see that $|\mathsf{adv}_H\rangle$ is a $O(2n \log(|\Sigma|) + \log(|\Sigma|^n)) = O(sn \log q)$-qubit state.

We now analyze the runtime of $\mathsf{BiasedYZ}$. First, observe that step 1 runs in time $O(sn \log q)$. To implement steps 2 and 5, note that $\mathrm{QFT}_\Sigma^{\otimes n}$ (and its inverse) can be implemented with total error at most $2^{-4\lambda}/4$ in

$\mathsf{BiasedYZ}_{\mathsf{Dec}_{C^\perp}}(|\mathsf{adv}_H\rangle, \mathbf{b})$:

1. If $|\mathsf{adv}_H\rangle \neq |\perp\rangle$, construct the state $(\bigotimes_{i=1}^n |\phi_{i,\mathbf{b}_i}\rangle)_\mathsf{A} \otimes |\psi\rangle_\mathsf{B}$ by re-arranging $|\mathsf{adv}_H\rangle$; else, return $\perp$.
2. Apply $(\mathsf{QFT}_\Sigma^{\otimes n})_\mathsf{A} \otimes (\mathsf{QFT}_\Sigma^{\otimes n})_\mathsf{B}$.
3. Controlled on register B, add the value of register B to register A.
4. Apply $\mathsf{Dec}_{C^\perp}$ to uncompute register B given the value of register A.
5. Apply $(\mathsf{QFT}_\Sigma^{-1,\otimes n})_\mathsf{A} \otimes \mathrm{id}_\mathsf{B}$ and measure register A to get an outcome $\mathbf{v} \in \Sigma^n$. Output $\mathbf{v}$.

Figure 1: Biased Yamakawa-Zhandry Algorithm $\mathsf{BiasedYZ}_{\mathsf{Dec}_{C^\perp}}(|\mathsf{adv}_H\rangle, \mathbf{b})$

$\mathrm{poly}(n, s, \log q, \log 2^{4\lambda}) = \mathrm{poly}(n, s, \log q, \lambda)$ time. Finally, step 3 consists of adding in Σ, which can be done in $\mathrm{poly}(n, s, \log q)$ time, and step 4 takes time T_{Dec} by definition. In total, we have that BiasedYZ runs in time $O(sn \log q) + \mathrm{poly}(n, s, \log q, \lambda) + \mathrm{poly}(n, s, \log q) + T_{\mathsf{Dec}} = \mathrm{poly}(n, s, \log q, \lambda) + T_{\mathsf{Dec}}$.

We finish by analyzing the correctness of BiasedYZ. For fixed $(i, b) \in [n] \times \{0, 1\}$, $|\phi_{i,b}\rangle \neq |\perp\rangle$ with probability at least $1 - (1-p)^{|\Sigma|}$ (as long as at least one symbol hashes to b under $H(i, \cdot)$). By a union bound, $|\mathsf{adv}_H\rangle \neq |\perp\rangle$ with probability at least $1 - 2n(1-p)^{|\Sigma|}$. Thus, using the definition of V and W from Section 5.2, we will assume for the remainder of the proof that $|\mathsf{adv}_H\rangle = \sum_{\mathbf{u},\mathbf{e} \in \Sigma^n} V(\mathbf{u}) W(\mathbf{e}) |\mathbf{u}\rangle |\mathbf{e}\rangle$. Then, by Lemma 5.17 and Corollary 5.20, we have that for all $\mathbf{b} \in S$,

$$\mathop{\mathbb{E}}_{H \leftarrow \mathsf{Bias}_{n,p,\Sigma}} \left[\sum_{(\mathbf{u},\mathbf{e}) \in \mathsf{BAD}} \left| \widehat{V}(\mathbf{u}) \cdot \widehat{W}^{H,\mathbf{b}}(\mathbf{e}) \right|^2 \right] = \mathop{\mathbb{E}}_{H \leftarrow \mathsf{Bias}_{n,p,\Sigma}} \left[\sum_{\mathbf{e} \in \mathcal{B}} \left| \widehat{W}^{H,\mathbf{b}}(\mathbf{e}) \right|^2 \right] \leq \mu(\lambda)$$

and

$$\mathop{\mathbb{E}}_{H \leftarrow \mathsf{Bias}_{n,p,\Sigma}} \left[\sum_{\mathbf{z} \in \Sigma^n} \left| \sum_{\substack{(\mathbf{u},\mathbf{e}) \in \mathsf{BAD}: \\ \mathbf{u}+\mathbf{e}=\mathbf{z}}} \widehat{V}(\mathbf{u}) \cdot \widehat{W}^{H,\mathbf{b}}(\mathbf{e}) \right|^2 \right] = \mathop{\mathbb{E}}_{H \leftarrow \mathsf{Bias}_{n,p,\Sigma}} \left[\sum_{\mathbf{z} \in \Sigma^n} \left| \sum_{\substack{\mathbf{u} \in C^\perp, \mathbf{e} \in \mathcal{B}: \\ \mathbf{u}+\mathbf{e}=\mathbf{z}}} \widehat{V}(\mathbf{u}) \cdot \widehat{W}^{H,\mathbf{b}}(\mathbf{e}) \right|^2 \right] \leq \mu(\lambda)\,.$$

Fixing $\mathbf{b} \in S$, Markov's inequality and the union bound implies that H satisfies both

$$\sum_{(\mathbf{u},\mathbf{e}) \in \mathsf{BAD}} \left| \widehat{V}(\mathbf{u}) \cdot \widehat{W}^{H,\mathbf{b}}(\mathbf{e}) \right|^2 \leq \mu(\lambda)^{1/2} \quad \text{and} \quad \sum_{\mathbf{z} \in \Sigma^n} \left| \sum_{(\mathbf{u},\mathbf{e}) \in \mathsf{BAD}: \mathbf{u}+\mathbf{e}=\mathbf{z}} \widehat{V}(\mathbf{u}) \cdot \widehat{W}^{H,\mathbf{b}}(\mathbf{e}) \right|^2 \leq \mu(\lambda)^{1/2}$$

with probability at least $1 - 2\mu(\lambda)^{1/2}$ over $\mathsf{Bias}_{n,p,\Sigma}$. For these H, by Lemma 5.10,

$$\Bigg\| \left(\mathrm{QFT}_\Sigma^{-1,\otimes n} \otimes \mathrm{id}\right) \cdot U_{\mathsf{Dec}_{C_\lambda^\perp}} \cdot U_{\mathsf{add}} \cdot \left(\mathrm{QFT}_\Sigma^{\otimes n} \otimes \mathrm{QFT}_\Sigma^{\otimes n}\right) \bigotimes_{i=1}^n |\phi_{i,\mathbf{b}_i}\rangle \otimes |\psi\rangle \\ - |\Sigma|^{n/2} \sum_{\mathbf{z} \in \Sigma^n} (V(\mathbf{z}) \cdot W^{H,\mathbf{b}}(\mathbf{z})) |\mathbf{z}\rangle |0\rangle \Bigg\| \leq 2\mu(\lambda)^{1/4}\,.$$

Since $|\mathsf{adv}_H\rangle \neq |\perp\rangle$ and $2\mu(\lambda)^{1/4} < 1$, the state $|\mathsf{tgt}\rangle = |\Sigma|^{n/2} \sum_{\mathbf{z} \in \Sigma^n} (V(\mathbf{z}) \cdot W^{H,\mathbf{b}}(\mathbf{z})) |\mathbf{z}\rangle$ is nonzero. But measuring (the normalization of) $|\mathsf{tgt}\rangle$ in the standard basis always produces vectors $\mathbf{v}$ such that $(\mathbf{b}, \mathbf{v}) \in R_{C,H}$. Thus, by Corollary 5.11, the output of BiasedYZ will be a vector $\mathbf{v}$ such that $(\mathbf{b}, \mathbf{v}) \in R_{C_\lambda,H}$ with probability at least $1 - 16\mu(\lambda)^{1/4} - 2^{-4\lambda}$ (where the $2^{-4\lambda}$ term comes from approximating the QFT).

We conclude that with probability $1 - (2n(1-p)^{|\Sigma|} + 2\mu(\lambda)^{1/2})$ over $\mathsf{Bias}_{n,p,\Sigma}$, BiasedYZ succeeds with probability at least $1 - 16\mu(\lambda)^{1/4} - 2^{-4\lambda}$ for any given $\mathbf{b} \in S$, as desired. □

Finally, we show that our choice of code satisfies the required conditions of Theorem 5.22.

Corollary 5.23. *For each security parameter $\lambda \in \mathbb{N}$, define the code $C_\lambda = \mathsf{Mult}_{s,\mathbb{F}_q,k}$, where $\lambda^5 < q \leq 2\lambda^5$ is a prime, $k = \lambda^3$, and $s = \lambda$. Then, there exists an efficient/uniform quantum algorithm* BiasedYZ *and a family of* $\mathrm{poly}(\lambda)$*-qubit states $\{|\mathsf{adv}_H\rangle\}_H$ such that the following holds for sufficiently large λ, for all strings $x \in \{0,1\}^\lambda$,*

$$\Pr_{H \leftarrow \mathsf{Bias}_{q,1/\lambda^4,\mathbb{F}_q^s}}\left[\Pr[(x\|0^{q-\lambda},\mathbf{v}) \in R_{C_\lambda,H} : \mathbf{v} \leftarrow \mathsf{BiasedYZ}(|\mathsf{adv}_H\rangle, x)] \geq 1 - 2^{-\lambda}\right] \geq 1 - 2^{-2\lambda}.$$

Proof. We begin by arguing that there exists a deterministic algorithm $\mathtt{Dec}_{C_\lambda^\perp}$ and function μ that for sufficiently large λ satisfies $\mu(\lambda) \leq 2^{-8\lambda}$, such that $T_{\mathtt{Dec}} = \mathrm{poly}(\lambda)$ and $(C_\lambda, \mathtt{Dec}_{C_\lambda^\perp})$ is $(\frac{1}{\lambda^4}, \mu, \{0,1\}^\lambda \times 0^{q-\lambda})$-good. Define the subset $\mathcal{G} := \left\{\mathbf{e} \in \Sigma^q : \mathsf{hw}(\mathbf{e}_{[\lambda+1:q]}) \leq \frac{11}{\lambda^4}(q-\lambda)\right\}$. By the Chernoff bound (Lemma 4.9), for any $x \in \{0,1\}^\lambda$,

$$\Pr_{\mathbf{e}\leftarrow\mathcal{D}_{1/\lambda^4,x\|0^{q-\lambda}}}[\mathbf{e} \notin \mathcal{G}] = \Pr_{\mathbf{e}\leftarrow\mathcal{D}_{1/\lambda^4,x\|0^{q-\lambda}}}\left[\mathsf{hw}(\mathbf{e}_{[\lambda+1:q]}) > 11 \cdot \frac{(q-\lambda)}{\lambda^4}\right] \leq e^{-\frac{100\lambda}{12}} \leq 2^{-8\lambda},$$

so it suffices to consider decoding only errors $\mathbf{e} \in \mathcal{G}$. Theorem 4.19 implies that for all $\mathbf{e} \in \mathcal{G}$ (assuming $\lambda \geq 50$),

$$\mathsf{hw}(\mathbf{e}) = \mathsf{hw}(\mathbf{e}_{[1:\lambda]}) + \mathsf{hw}(\mathbf{e}_{[\lambda+1:q]}) < \lambda + \frac{11}{\lambda^4}(q-\lambda) \leq 25\lambda \leq \lambda^2/2 \leq \mathrm{dist}(C_\lambda^\perp)/2\,.$$

It therefore suffices to *uniquely* decode $C_\lambda^\perp$ in a deterministic and efficient manner. We set μ to be the maximum probability (across all x) that unique decoding for $C_\lambda^\perp$ fails with error distribution $\mathcal{D}_{1/\lambda^4,x\|0^{q-\lambda}}$; thus, by our previous argument, $\mu(\lambda) \leq 2^{-8\lambda}$ as long as $\lambda \geq 50$.

By Theorem A.5, we know that $C_\lambda^\perp = \mathsf{GM}_{s,\mathbb{F}_q}(U_1,\ldots,U_q;1,\ldots,q;sq-k)$. From Lemma A.3, it is easy to see that $A_{i,j}(X)$ can be computed in $\mathrm{poly}(s,q) = \mathrm{poly}(\lambda)$ time, and thus $a_{i,j}$ (and consequently U_i and U_i^{-1}) can be computed in $\mathrm{poly}(\lambda)$ time. Finally, it remains to efficiently uniquely decode $\mathsf{Mult}_{s,\mathbb{F}_q,sq-k}$, which we can do deterministically in $\mathrm{poly}(s,q) = \mathrm{poly}(\lambda)$ time [Nie01, KSY14, Kop15].[5]

Applying Theorem 5.22 gives a family of $\mathrm{poly}(\lambda)$-qubit states $\{|\mathsf{adv}_H\rangle\}_H$ and a $\mathrm{poly}(\lambda)$-time uniform algorithm $\mathcal{B}$ such that for $\lambda \geq 50$ and all strings $x \in \{0,1\}^\lambda$,

$$\Pr_{H\leftarrow\mathsf{Bias}_{q,1/\lambda^4,\Sigma}}\left[\Pr[(x\|0^{q-\lambda},\mathbf{v}) \in R_{C_\lambda,H} : \mathbf{v} \leftarrow \mathcal{B}(|\mathsf{adv}_H\rangle, x\|0^{q-\lambda})] \geq 1 - 2^{-2\lambda+4} - 2^{-4\lambda}\right] \geq 1 - 2q(1-p)^{|\Sigma|} - 2^{-4\lambda+1}$$

$$\implies \Pr_{H\leftarrow\mathsf{Bias}_{q,1/\lambda^4,\mathbb{F}_q^s}}\left[\Pr[(x\|0^{q-\lambda},\mathbf{v}) \in R_{C_\lambda,H} : \mathbf{v} \leftarrow \mathcal{B}(|\mathsf{adv}_H\rangle, x\|0^{q-\lambda})] \geq 1 - 2^{-\lambda}\right] \geq 1 - 2^{-2\lambda}.$$

The algorithm BiasedYZ simply runs $\mathcal{B}(|\mathsf{adv}_H\rangle, x\|0^{q-\lambda})$ given advice $|\mathsf{adv}_H\rangle$ and input $x \in \{0,1\}^\lambda$. □

We therefore have the following corollary by a simple union bound over all $x \in \{0,1\}^\lambda$.

Corollary 5.24. *For each security parameter $\lambda \in \mathbb{N}$, define the code $C_\lambda = \mathsf{Mult}_{s,\mathbb{F}_q,k}$, where $\lambda^5 < q \leq 2\lambda^5$ is a prime, $k = \lambda^3$, and $s = \lambda$. Then, there exists an efficient/uniform quantum algorithm* BiasedYZ *and a family of* $\mathrm{poly}(\lambda)$*-qubit states $\{|\mathsf{adv}_H\rangle\}_H$ such that for sufficiently large λ,*

$$\Pr_{H\leftarrow\mathsf{Bias}_{q,1/\lambda^4,\mathbb{F}_q^s}}\left[\forall x \in \{0,1\}^\lambda, \Pr[(x\|0^{q-\lambda},\mathbf{v}) \in R_{C_\lambda,H} : \mathbf{v} \leftarrow \mathsf{BiasedYZ}(|\mathsf{adv}_H\rangle, x)] \geq 1 - 2^{-\lambda}\right] \geq 1 - 2^{-\lambda}\,.$$

6 Separating QMA from QCMA

For the rest of the paper, we fix a code family $C_\lambda := \mathsf{Mult}_{s,\mathbb{F}_q,k}$ for $\lambda^5 < q \leq 2\lambda^5$ a prime,[6] $k = \lambda^3$, and $s = \lambda$. For any subset $E \subseteq \{0,1\}^\lambda \times \Sigma^q$ and function $H : [q] \times \Sigma \to \{0,1\}$, we define the oracle

$$O[H,E](x,\mathbf{v}) = \begin{cases} 1 & \text{if } (x\|0^{q-\lambda},\mathbf{v}) \in R_{C_\lambda,H} \wedge (x,\mathbf{v}) \in E, \\ 0 & \text{otherwise.} \end{cases}$$

[5] In fact, a simple extension of the Berlekamp-Welch algorithm [WB86] gives efficient unique decoding for univariate multiplicity codes.
[6] To be concrete, we can take q to be the smallest prime larger than λ^5 (which is always at most $2\lambda^5$).

Our proofs in this section are fairly standard and follow [BHNZ25], but we include them for completeness.

We begin by defining an oracle-input problem based on the code intersection subset size checking problem. Note that our NO instances are slightly different than those considered in [LLPY23, BDK24], which will affect the QCMA lower bound, but not the QMA containment.

Definition 6.1 (YES and NO instances of the code intersection subset size problem)**.** *Let* Good *denote the set of functions* $H : [q] \times \mathbb{F}_q^s \to \{0,1\}$ *such that the algorithm in Corollary 5.24 succeeds on all* $x \in \{0,1\}^\lambda$ *with probability at least* $2/3$*. Our oracle-input separation between* QMA *and* QCMA *involves distinguishing between* $O[H, E]$ *with* H *and* E *being the following:*

1. *YES instances:* $H \in \mathsf{Good}$ *and* $E = \{0,1\}^\lambda \times \Sigma^q$*.*
2. *NO instances:* $H \in \mathsf{Good}$ *and subsets* $E \subseteq F \times \Sigma^q \subseteq \{0,1\}^\lambda \times \Sigma^q$ *such that* $|F| \leq 2^\lambda/3$*.*

Remark 6.2. *One can also embed* $O[H, E] : \{0,1\}^\lambda \times \Sigma^q \to \{0,1\}$ *into a binary input domain oracle with* $\lambda + q \log |\Sigma| \leq \lambda^7$*-bit inputs, in such a way that for sufficiently large* $\lambda \geq \lambda_0$ *(where* $\lambda_0 \in \mathbb{N}$ *is some constant), there is at most one security parameter associated with each input length.*

6.1 The QMA Proof System

We first show that there is a QMA proof system that distinguishes between YES and NO instances of the code intersection subset size checking problem, as defined in Definition 6.1.

Lemma 6.3 (A QMA proof system)**.** *There exists a polynomial-time uniform quantum query algorithm* V *which makes one query to the oracle* $O[H, E]$*, such that for sufficiently large* λ*, the following holds:*

1. **Completeness.** *For all* $H \in \mathsf{Good}$*, when* $E = \{0,1\}^\lambda \times \Sigma^q$*, there exists a* $\mathrm{poly}(\lambda)$*-qubit state* $|\mathsf{adv}_H\rangle$ *such that*

$$\Pr[V^{O[H,E]}(|\mathsf{adv}_H\rangle) = 1] \geq \frac{2}{3}.$$

2. **Soundness.** *For all* H*, sets* $E \subseteq F \times \Sigma^q \subseteq \{0,1\}^\lambda \times \Sigma^q$ *where* $|F| \leq 2^\lambda/3$*, and quantum states* $\left|\mathsf{adv}_H^*\right\rangle$*,*

$$\Pr[V^{O[H,E]}(\left|\mathsf{adv}_H^*\right\rangle) = 1] \leq \frac{1}{3}.$$

Proof. By definition, for $H \in \mathsf{Good}$ as defined in Corollary 5.24, there exists a $\mathrm{poly}(\lambda)$-qubit state $|\mathsf{adv}_H\rangle$ and efficient algorithm $\mathsf{BiasedYZ}$ such that for any $x \in \{0,1\}^\lambda$,

$$\Pr_{\mathsf{BiasedYZ}}[(x \| 0^{q-\lambda}, \mathbf{v}) \in R_{C_\lambda, H} : \mathbf{v} \leftarrow \mathsf{BiasedYZ}(|\mathsf{adv}_H\rangle, x)] \geq 1 - 2^{-\lambda} \geq \frac{2}{3}.$$

The verifier V operates as follows: it samples a uniformly random $x \in \{0,1\}^\lambda$ and runs $\mathbf{v} \leftarrow \mathsf{BiasedYZ}(|\mathsf{adv}_H\rangle, x)$. Finally, V queries O at $(x, \mathbf{v})$ and returns the output of O. The efficiency/uniformity of V follows from the efficiency of $\mathsf{BiasedYZ}$ and the fact that V makes one oracle query.

We now argue completeness and soundness. If $E = \{0,1\}^\lambda \times \Sigma^q$, then

$$\Pr[V^{O[H,E]}(|\mathsf{adv}_H\rangle) = 1] = \Pr_{\mathsf{BiasedYZ}, x}[(x \| 0^{q-\lambda}, \mathbf{v}) \in R_{C_\lambda, H} : \mathbf{v} \leftarrow \mathsf{BiasedYZ}(|\mathsf{adv}_H\rangle, x)] \geq \frac{2}{3}.$$

On the other hand, if $E \subseteq F \times \Sigma^q \subseteq \{0,1\}^\lambda \times \Sigma^q$ and $|F| \leq 2^\lambda/3$, then V will never output 1 if $x \notin F$. Thus, for all H and quantum states $\left|\mathsf{adv}_H^*\right\rangle$,

$$\Pr[V^{O[H,E]}(\left|\mathsf{adv}_H^*\right\rangle) = 1] \leq \Pr_{x \leftarrow \{0,1\}^\lambda}[x \in F] \leq \frac{1}{3}. \qquad \square$$

6.2 Non-Existence of QCMA Proof Systems

We begin by showing that any QCMA verifier/algorithm can be turned into a very good hash value guesser.

Lemma 6.4 (Good guessers from QCMA algorithms)**.** *Assume there exists a* QCMA *algorithm $\mathcal{A}$ such that for instances of size λ, $\mathcal{A}$ takes a $t(\lambda)$-bit witness and makes $Q(\lambda)$ oracle queries. Then for all $\ell \leq 2^{\lambda/3}$, there exists a algorithm* Guesser *which makes* no *queries such that for any $H \in$ Good,*

$$\Pr\left[\begin{matrix}\forall i \neq j, (x_i, \mathbf{v}_i) \neq (x_j, \mathbf{v}_j) \\ \wedge\, (x_i \| 0^{q-\lambda}, \mathbf{v}_i) \in R_{C_\lambda, H}\end{matrix} : \{(x_i, \mathbf{v}_i)\}_{i=1}^{\ell} \leftarrow \mathsf{Guesser}(1^\ell)\right] \geq 2^{-t(\lambda)} \cdot \left(\frac{1}{144Q(\lambda)^2}\right)^\ell .$$

Proof. The algorithm $\mathcal{A}^{\mathcal{O}[H,E]}$ can be thought of starting from a state $|w, 0\rangle$ and applying a sequence of unitaries $V_0, \ldots, V_Q$ interlaced with queries to $\mathcal{O}[H, E]$ before measuring the first qubit in the standard basis. The state of the algorithm right before its final measurement is then given by

$$V_Q \cdot \mathcal{O}[H, E] \cdot V_{Q-1} \cdot \mathcal{O}[H, E] \ldots \mathcal{O}[H, E] \cdot V_0 \, |w, 0\rangle \ .$$

Let Guesser be the algorithm described in Figure 2 which outputs ℓ tuples of codewords and hash values.

Guesser(1^ℓ)**:**

1. Sample a random $w \in \{0,1\}^t$ and initialize $\Delta_0 = \emptyset$.
2. For $i \in [\ell]$:
   (a) Sample $j \leftarrow \{0, \ldots, Q-1\}$ uniformly randomly.
   (b) Compute the state $V_j \mathcal{O}_{\Delta_{i-1}} V_{j-1} \ldots \mathcal{O}_{\Delta_{i-1}} V_0 \, |w, 0\rangle$, where $\mathcal{O}_{\Delta_{i-1}}$ is the oracle unitary defined by

$$|x, \mathbf{v}\rangle \, |y\rangle \, |z\rangle \mapsto |x, \mathbf{v}\rangle \left|y \oplus f_{\Delta_{i-1}}(x, \mathbf{v})\right\rangle |z\rangle \, , \text{ where } f_{\Delta_{i-1}}(x, \mathbf{v}) := \begin{cases} 1 & \text{if } (x, \mathbf{v}) \in \Delta_{i-1}, \\ 0 & \text{otherwise.} \end{cases}$$

   (c) Measure the first register in the standard basis for output $(x, \mathbf{v})$ and update $\Delta_i := \Delta_{i-1} \cup \{(x, \mathbf{v})\}$.
3. Output Δ_ℓ.

Figure 2: The Hash Value Guesser, given a successful QCMA verifier for the code intersection subset size problem.

Let G be the event that the witness w sampled is a good witness for H, and let E_i be the event that the i'th round of Guesser appends a tuple $(x, \mathbf{v}) \notin \Delta_{i-1}$ such that $(x \| 0^{q-\lambda}, \mathbf{v}) \in R_{C_\lambda, H}$.

Claim 6.5. $\forall i \in [\ell]$, $\Pr[E_i | E_{i-1} \wedge \ldots \wedge E_1 \wedge G] \geq \frac{1}{144Q(\lambda)^2}$.

Proof. Fix an index $1 \leq i \leq \ell$. Observe that conditioned on $E_1 \wedge \ldots \wedge E_{i-1}$ occurring, this means that Δ_{i-1} consists of $i-1$ distinct tuples $\{(x_j, \mathbf{v}_j)\}_{j=1}^{i-1}$ such that $(x_j \| 0^{q-\lambda}, \mathbf{v}_j) \in R_{C_\lambda, H}$ for all $1 \leq j \leq i-1$.

Since $\mathcal{O}_{\Delta_{i-1}} = \mathcal{O}[H, \Delta_{i-1}]$ corresponds to a NO instance (as $|\Delta_{i-1}| = i-1 < \ell \leq 2^\lambda/3$) while $\mathcal{O}[H, \{0,1\}^\lambda \times \Sigma^q]$ corresponds to a YES instance, by the completeness and soundness of the QCMA algorithm $\mathcal{A}$ and the fact that the event G implies we have a good witness, Theorem 4.8 implies that the query mass on the inputs where the two oracles differ must be at least $((2/3 - 1/3)/4)^2/Q = 1/144Q$.

As $\mathcal{O}_{\Delta_{i-1}}$ and $\mathcal{O}[H, \{0,1\}^\lambda \times \Sigma^q]$ differ precisely on inputs $(x, \mathbf{v}) \notin \Delta_{i-1}$ where $(x \| 0^{q-\lambda}, \mathbf{v}) \in R_{C_\lambda, H}$, it follows that measuring a random query of $\mathcal{A}$ produces a good tuple $(x, \mathbf{v})$ with probability at least $\frac{1}{Q} \cdot \frac{1}{144Q} = \frac{1}{144Q^2}$. □

Observing that $\Pr[G] \geq 2^{-t(\lambda)}$ as there is always at least one good witness for any H, we conclude that

$$\Pr[E_1 \wedge \ldots \wedge E_\ell] \geq \Pr[G] \cdot \prod_{i=1}^{\ell} \Pr[E_i | E_{i-1} \wedge \ldots \wedge E_1 \wedge G] \geq 2^{-t(\lambda)} \cdot \left(\frac{1}{144Q(\lambda)^2}\right)^\ell . \qquad \square$$

Separately, we can show the following *upper bound* on the success probability of Guesser. The bound follows from the fact that Guesser is not making any queries to the oracle, and thus knows nothing about H.

Lemma 6.6 (Guessing probability upper bound)**.** *For sufficiently large λ, the following holds: fix any algorithm* Guesser *which makes no oracle queries. Then, for all $\ell \leq 2^\lambda$,*

$$\Pr_{H \leftarrow \mathsf{Bias}_{q,1/\lambda^4,\mathbb{F}_q^s}}\left[\begin{matrix}\forall i \neq j, (x_i,\mathbf{v}_i) \neq (x_j,\mathbf{v}_j) \\ \wedge\, (x_i \| 0^{q-\lambda}, \mathbf{v}_i) \in R_{C_\lambda,H}\end{matrix} : \{(x_i,\mathbf{v}_i)\}_{i=1}^{\ell} \leftarrow \mathsf{Guesser}(1^\ell)\right] \leq \left(1-\frac{1}{\lambda^4}\right)^{\lambda^5 \ell/2}.$$

Proof. Consider any output $(x_1,\mathbf{v}_1,\ldots,x_\ell,\mathbf{v}_\ell)$ of Guesser. First, observe that if $(x_i,\mathbf{v}_i)$ are distinct and $H(\mathbf{v}_i) = x_i\|0^{q-\lambda}$, then we must have distinct $\mathbf{v}_i$ or else Guesser will fail. Since we require $\mathbf{v}_i \in C_\lambda$ for all $i \in [\ell]$, we now consider the sets $S_j := \{\sigma \in \Sigma \mid \exists i \in [\ell] : (\mathbf{v}_i)_j = \sigma\}$ for all $j \in [q]$.

By definition, for all $i \in [\ell]$ and $j \in [q]$, $(\mathbf{v}_i)_j \in S_j$, so if we think of $\{S_j\}_{j=1}^q$ as input lists, the output list for the code C_λ must contain $\mathbf{v}_i \in [\ell]$ and thus $|\{\mathbf{v} \in C_\lambda : \forall j \in [q], \mathbf{v}_j \in S_j\}| \geq \ell$.

Corollary 4.18 thus implies that $\frac{1}{q}\sum_j |S_j| \geq \ell/2$ if λ is sufficiently large. In order for Guesser to succeed, it must correctly guess the output of H on all symbols in $\bigcup_{i=1}^q S_i$ which contain at least $\sum_j |S_j| \geq \frac{q\ell}{2} = \lambda^5\ell/2$ distinct points. Since we sample $H \leftarrow \mathsf{Bias}_{q,1/\lambda^4,\mathbb{F}_q^s}$, this occurs with probability at most $\left(1-\frac{1}{\lambda^4}\right)^{\lambda^5\ell/2}$. □

We can combine the upper bound and lower bound to conclude that any QCMA algorithm $\mathcal{A}$ for the code intersection subset size problem must misclassify some YES or NO instance.

Lemma 6.7. *For all constants $a > 0$ and functions $Q(\lambda)$, $t(\lambda)$ that satisfy $Q(\lambda) \leq a\lambda^a$, $t(\lambda) \leq a\lambda^a$ for sufficiently large λ. Then for sufficiently large λ, for all quantum query algorithms $\mathcal{A}$ which take a classical witness of length $t(\lambda)$ and make $Q(\lambda)$ queries to the oracle $O[H,E]$ of size λ, there exists an oracle $O[H^*,E^*]$ of size λ such that $H^* \in$ Good and*

1. *either $E^* = \{0,1\}^\lambda \times \Sigma^q$, but for all witnesses w of length $t(\lambda)$,*

$$\Pr[\mathcal{A}^{O[H^*,E^*]}(w) = 1] < \frac{2}{3},$$

2. *or $E^* \subseteq F^* \times \Sigma^q \subseteq \{0,1\}^\lambda \times \Sigma^q$ where $|F^*| \leq 2^\lambda/3$, but there exists a witness $\widetilde{w}$ of length $t(\lambda)$ such that*

$$\Pr[\mathcal{A}^{O[H^*,E^*]}(\widetilde{w}) = 1] > \frac{1}{3}.$$

Proof. Suppose for the sake of contradiction that $\mathcal{A}$ properly classifies all YES and NO instances. Setting $\ell = t \ll 2^\lambda/3$, we can apply Corollary 5.24 and Lemma 6.4, yielding a guesser Guesser where

$$\begin{aligned}
&\Pr_{H \leftarrow \mathsf{Bias}_{q,1/\lambda^4,\mathbb{F}_q^s}}\left[\begin{matrix}\forall i \neq j, (x_i,\mathbf{v}_i) \neq (x_j,\mathbf{v}_j) \\ \wedge\, (x_i \| 0^{q-\lambda}, \mathbf{v}_i) \in R_{C_\lambda,H}\end{matrix} : \{(x_i,\mathbf{v}_i)\}_{i=1}^{t} \leftarrow \mathsf{Guesser}(1^t)\right] \\
&\geq \Pr_{H \leftarrow \mathsf{Bias}_{q,1/\lambda^4,\mathbb{F}_q^s}}\left[\begin{matrix}\forall i \neq j, (x_i,\mathbf{v}_i) \neq (x_j,\mathbf{v}_j) \\ \wedge\, (x_i \| 0^{q-\lambda}, \mathbf{v}_i) \in R_{C_\lambda,H}\end{matrix} : \{(x_i,\mathbf{v}_i)\}_{i=1}^{t} \leftarrow \mathsf{Guesser}(1^t) \,\middle|\, H \in \mathsf{Good}\right] \cdot \Pr_{H \leftarrow \mathsf{Bias}_{q,1/\lambda^4,\mathbb{F}_q^s}}[H \in \mathsf{Good}] \\
&\geq 2^{-t(\lambda)} \cdot \left(\frac{1}{144Q(\lambda)^2}\right)^{t(\lambda)} \cdot (1-2^{-\lambda}) \geq \left(\frac{1}{576Q(\lambda)^2}\right)^{t(\lambda)}.
\end{aligned}$$

But this is impossible as Lemma 6.6 implies that the success probability of Guesser is at most

$$\left(1-\frac{1}{\lambda^4}\right)^{\lambda^5 t(\lambda)/2} \leq (e^{-\lambda/2})^{t(\lambda)} \ll \left(\frac{1}{576Q(\lambda)^2}\right)^{t(\lambda)}.$$ □

A straightforward diagonalization argument thus gives us our desired separation (see Section §B for details).

Theorem 6.8. *There exists a classical oracle $\mathcal{O} : \{0,1\}^* \to \{0,1\}$ such that* $\mathsf{QMA}^{\mathcal{O}} \cap \mathsf{AM}^{\mathcal{O}} \not\subseteq \mathsf{QCMA}^{\mathcal{O}}$.[7]

[7] We note that this separation can be easily strengthened to $\mathsf{QMA}^{\mathcal{O}} \cap \mathsf{SBP}^{\mathcal{O}} \not\subseteq \mathsf{QCMA}^{\mathcal{O}}$, as with most set-approximation-flavored oracles.

7 Separating BQP/qpoly from BQP/poly

We begin by proving the main technical result of this section, which is a search-like separation between BQP/qpoly and BQP/poly. In particular, we prove that given as input an oracle $O[H, E]$ for $E = \{0,1\}^\lambda \times \Sigma^q$ and H sampled from $\mathsf{Bias}_{q,1/\lambda^4,\mathbb{F}_q^s}$, and $x \in \{0,1\}^\lambda$, the problem of finding a codeword $\mathbf{v}$ whose hash values correspond to x is in BQP/qpoly (where the advice is allowed to depend on the oracle, but not on x) but not BQP/poly.

Lemma 7.1. *For all security parameters $\lambda \in \mathbb{N}$, let $\lambda^5 < q < 2\lambda^5$ be a prime, $k = \lambda^3$, and $s = \lambda$, and define the code $C_\lambda = \mathsf{Mult}_{s,\mathbb{F}_q,k}$. In addition, for all λ, define the set $E_\lambda := \{0,1\}^\lambda \times \Sigma^q$, where $\Sigma = \mathbb{F}_q^s$. Then the following hold:*

1. *There is an polynomial-time uniform quantum query algorithm $\mathcal{A}$, such that for all oracles O there exists a family of $\mathrm{poly}(\lambda)$-qubit quantum advice states (depending only on the oracle) $\{|z_O\rangle\}_O$ such that*

$$\Pr_{H\leftarrow\mathsf{Bias}_{q,1/\lambda^4,\mathbb{F}_q^s}}[\forall x \in \{0,1\}^\lambda, \Pr[(x\|0^{q-\lambda},\mathbf{v}) \in R_{C_\lambda,H} : \mathbf{v} \leftarrow \mathcal{A}^{O[H,E_\lambda]}(x,|z_O\rangle)] \geq 1-\mathrm{negl}(\lambda)] \geq 1-\mathrm{negl}(\lambda).$$

2. *For all unbounded-time quantum algorithms $\mathcal{B}$ that make $Q(\lambda) = \mathrm{poly}(\lambda)$ oracle queries to O, and all families of $t(\lambda) = \mathrm{poly}(\lambda)$-bit classical advice strings (depending only on the oracle) $\{z_O\}_O$,*

$$\Pr_{H\leftarrow\mathsf{Bias}_{q,1/\lambda^4,\mathbb{F}_q^s},x\leftarrow\{0,1\}^\lambda}[(x\|0^{q-\lambda},\mathbf{v}) \in R_{C_\lambda,H} : \mathbf{v} \leftarrow \mathcal{B}^{O[H,E_\lambda]}(x,z_O)] \leq \mathrm{negl}(\lambda).$$

Proof. Throughout this proof we will assume that λ is sufficiently large and argue with respect to asymptotics. To prove Item 1, we simply set the advice as $|z_O\rangle = |\mathsf{adv}_H\rangle$ from Corollary 5.24, and the algorithm just runs the algorithm in Corollary 5.24, which implies that

$$\Pr_{H\leftarrow\mathsf{Bias}_{q,1/\lambda^4,\mathbb{F}_q^s}}\left[\forall x \in \{0,1\}^\lambda, \Pr[(x\|0^{q-\lambda},\mathbf{v}) \in R_{C_\lambda,H} : \mathbf{v} \leftarrow \mathcal{A}(|z_O\rangle, x)] \geq 1-2^{-\lambda}\right] \geq 1-2^{-\lambda}.$$

We now move to proving Item 2. Suppose for the sake of contradiction that there exists a polynomial $p(\lambda)$, an adversary $\mathcal{B}$ which makes $Q(\lambda) = \mathrm{poly}(\lambda)$ oracle queries, and a family of $t(\lambda) = \mathrm{poly}(\lambda)$-bit classical advice $\{z_O\}_O$ such that for infinitely many λ,

$$\Pr_{H\leftarrow\mathsf{Bias}_{q,1/\lambda^4,\mathbb{F}_q^s},x\leftarrow\{0,1\}^\lambda}\left[(x\|0^{q-\lambda},\mathbf{v}) \in R_{C_\lambda,H} : \mathbf{v} \leftarrow \mathcal{B}^{O[H,E_\lambda]}(z_O,x)\right] \geq \frac{1}{p(\lambda)}.$$

Our goal will be to arrive at a contradiction by showing that this algorithm $\mathcal{B}$ implies a (too good) sampler for $R_{C_\lambda,H}$ that works for infinitely many choices of λ. Consider the Q-query algorithm $\mathcal{A}_1^{O[H,E_\lambda]}(z_O)$ which samples a random $x \leftarrow \{0,1\}^\lambda$, runs $\mathcal{B}^{O[H,E_\lambda]}(z_O,x)$ to get $\mathbf{v}$, and outputs $(x,\mathbf{v})$. By the definition of $\mathcal{B}$, we have

$$\Pr_{H\leftarrow\mathsf{Bias}_{q,1/\lambda^4,\mathbb{F}_q^s}}\left[(x\|0^{q-\lambda},\mathbf{v}) \in R_{C_\lambda,H} : (x,\mathbf{v}) \leftarrow \mathcal{A}_1^{O[H,E_\lambda]}(z_O)\right] \geq \frac{1}{p(\lambda)}, \tag{1}$$

for infinitely many λ. We denote with Λ the set of all λ for which this bound holds. At a very high level, we will first show that for all but finitely many $\lambda \in \Lambda$, $\mathcal{A}_1$'s queries are concentrated on very few points. Once we know that queries to $\mathcal{A}_1$ are concentrated on a couple points, replacing the real oracle with an oracle that only contains those few points will give rise to a sampler for many more points than are contained within the oracle itself.

For each function H and set $S \subseteq \{0,1\}^\lambda \times \Sigma^q$, we define $M_{H,S}$ as the query mass that $\mathcal{A}_1$ places on points which differ between $O[H,S]$ and $O[H,E]$. Let $\mathsf{SmallSet}_H$ be the event that there exists a list $L_H \subseteq \{0,1\}^\lambda \times \Sigma^q$ such that $|L_H| \leq 2^{\lambda/2}$ and $M_{H,\overline{L_H}} \leq \frac{1}{256p^2(\lambda)Q(\lambda)}$.

Claim 7.2. *Whenever $\mathsf{SmallSet}_H$ does not occur, for all $\ell \leq 2^{\lambda/2}$, there is an algorithm, $\mathsf{Guesser}$, which makes no queries to an oracle, and outputs a list of ℓ distinct points from $R_{C_\lambda,H}$ with probability at least $2^{-t} \cdot \left(\frac{1}{256p^2(\lambda)Q^2(\lambda)}\right)^\ell$.*

Proof. The algorithm $\mathsf{Guesser}$ is identical to the algorithm in Figure 2, except starting from $\mathcal{A}_1$ instead of a QCMA verifier for the code intersection subset size problem.

The proof follows similarly as well. Let X_i be the event that the i'th round of $\mathsf{Guesser}(1^\ell)$ outputs a tuple $(x, \mathbf{v}) \notin \Delta_{i-1}$ such that $(x\|0^{q-\lambda}, \mathbf{v}) \in R_{C_\lambda, H}$ and G be the event that the advice is guessed correctly. By assumption, and because $|\Delta_{i-1}| \leq 2^{\lambda/2}$, we have that $\mathcal{A}_1$ places at least $\frac{1}{256p^2(\lambda)Q(\lambda)}$ query mass on points which differ between $O[H, \Delta_{i-1}]$ and $O[H, E_\lambda]$. Therefore, we have that $\Pr[X_i | X_1 \wedge ... \wedge X_{i-1} \wedge G] \geq \frac{1}{256p^2(\lambda)Q^2(\lambda)}$.

Applying the chain rule, together with the fact that $\Pr[G] \geq 2^{-t(\lambda)}$, we get that the probability of sampling ℓ distinct points from $R_{C_\lambda, H}$ is at least

$$\Pr[X_1 \wedge ... \wedge X_\ell \wedge G] \geq 2^{-t(\lambda)} \cdot \left(\frac{1}{256p^2(\lambda)Q^2(\lambda)}\right)^\ell . \qquad \square$$

As a corollary, we have that for all but finitely many $\lambda \in \Lambda$, $\mathsf{SmallSet}_H$ must occur with high probability.

Claim 7.3. *There are only finitely many $\lambda \in \Lambda$ such that $\Pr_{H \leftarrow \mathsf{Bias}_{q,1/\lambda^4,\mathbb{F}_q^s}}[\mathsf{SmallSet}_H] < 1 - \frac{1}{4p(\lambda)}$.*

Proof. Assume for the sake of contradiction that there are infinitely many $\lambda \in \Lambda$ such that $\Pr[\mathsf{SmallSet}_H] < 1 - \frac{1}{4p(\lambda)}$. Whenever $\mathsf{SmallSet}_H$ does not occur, the previous claim gives us a sampler. Thus, when H is sampled from $\mathsf{Bias}_{q,1/\lambda^4,\mathbb{F}_q^s}$, the probability of the sampler outputting ℓ distinct points from $R_{C_\lambda, H}$ is at least

$$\Pr_{H \leftarrow \mathsf{Bias}_{q,1/\lambda^4,\mathbb{F}_q^s}}\left[\begin{matrix}\forall i \neq j, (x_i, \mathbf{v}_i) \neq (x_j, \mathbf{v}_j) \\ \wedge\, (x_i\|0^{q-\lambda}, \mathbf{v}_i) \in R_{C_\lambda, H}\end{matrix} : \{(x_i, \mathbf{v}_i)\}_{i=1}^\ell \leftarrow \mathsf{Guesser}(1^\ell)\right] \geq \frac{1}{4p} \cdot 2^{-t} \cdot \left(\frac{1}{256p^2(\lambda)Q^2(\lambda)}\right)^\ell .$$

Taking $\ell = \max\{t, \lambda\} \leq 2^{\lambda/2} \leq 2^\lambda$ gives a sampling success probability of

$$\frac{1}{4p(\lambda)} \cdot \left(\frac{1}{512p(\lambda)^2Q(\lambda)^2}\right)^\ell \geq \left(\frac{1}{2048p(\lambda)^3Q(\lambda)^2}\right)^\ell ,$$

which is a contradiction since Lemma 6.6 implies this success probability should be at most

$$\left(e^{-\lambda/2}\right)^\ell \ll \left(\frac{1}{2048p(\lambda)^3Q(\lambda)^2}\right)^\ell . \qquad \square$$

Now we know that $\mathsf{SmallSet}_H$ occurs with high probability for infinitely many $\lambda \in \Lambda$. For all H for which $\mathsf{SmallSet}_H$ occurs, we denote L_H^* to refer to any arbitrary set of size at most $2^{\lambda/2}$ such that $M_{H, \overline{L_H^*}} \leq \frac{1}{256p^2(\lambda)Q(\lambda)}$; if $\mathsf{SmallSet}_H$ does not occur, then we define $L_H^* := \emptyset$. For each function H, define the punctured oracle

$$O_H^*(x, \mathbf{v}) = \begin{cases} 1 & \text{if } (x\|0^{q-\lambda}, \mathbf{v}) \in R_{C_\lambda, H} \wedge (x, \mathbf{v}) \in L_H^*, \\ 0 & \text{otherwise.} \end{cases}$$

Then we have that, conditioned on λ being such that $\Pr_{H \leftarrow \mathsf{Bias}_{q,1/\lambda^4,\mathbb{F}_q^s}}[\mathsf{SmallSet}_H] \geq 1 - \frac{1}{4p(\lambda)}$,

$$\begin{aligned}
&\Pr_{H \leftarrow \mathsf{Bias}_{q,1/\lambda^4,\mathbb{F}_q^s}}\left[(x\|0^{q-\lambda}, \mathbf{v}) \in R_{C_\lambda, H} : (x, \mathbf{v}) \leftarrow \mathcal{A}_1^{O_H^*}(z_O) | \mathsf{SmallSet}_H\right] \\
&\geq \Pr_{H \leftarrow \mathsf{Bias}_{q,1/\lambda^4,\mathbb{F}_q^s}}\left[(x\|0^{q-\lambda}, \mathbf{v}) \in R_{C_\lambda, H} : (x, \mathbf{v}) \leftarrow \mathcal{A}_1^{O[H,E]}(z_O) | \mathsf{SmallSet}_H\right] - \frac{1}{4p(\lambda)} \\
&\geq \Pr_{H \leftarrow \mathsf{Bias}_{q,1/\lambda^4,\mathbb{F}_q^s}}\left[(x\|0^{q-\lambda}, \mathbf{v}) \in R_{C_\lambda, H} : (x, \mathbf{v}) \leftarrow \mathcal{A}_1^{O[H,E]}(z_O) \wedge \mathsf{SmallSet}_H\right] - \frac{1}{4p(\lambda)} \\
&\geq \frac{1}{p(\lambda)} - \frac{1}{2p(\lambda)} = \frac{1}{2p(\lambda)} .
\end{aligned}$$

In the first line, we use the hybrid lemma (Theorem 4.8) combined with our bound on the query mass of $\mathcal{A}_1$ outside of L_H. In the second line, we use the definition of conditional probability, together with the fact that

$\Pr[\mathsf{SmallSet}_H] \leq 1$. We conclude by using a union bound, together with the fact that $\Pr[\mathsf{SmallSet}_H] \geq 1 - \frac{1}{4p(\lambda)}$ and the fact that the probability of $\mathcal{A}_1$ sampling a point in $R_{C_\lambda,H}$ is at least $\frac{1}{p(\lambda)}$ by assumption.

To arrive at a contradiction we construct yet *another* sampler. By the definition of $\mathcal{A}_1$, running it produces a uniformly random $x \in \{0,1\}^\lambda$, so there exists an algorithm $\mathcal{A}_2$ which, on input 1^ℓ, runs $\mathcal{A}_1$ ℓ times and satisfies

$$\Pr_{H \leftarrow \mathsf{Bias}_{q,1/\lambda^4,\mathbb{F}_q^s}}\left[\begin{matrix}\forall i \neq j, (x_i,\mathbf{v}_i) \neq (x_j,\mathbf{v}_j) \\ \wedge\, (x_i \| 0^{q-\lambda}, \mathbf{v}_i) \in R_{C_\lambda,H}\end{matrix} : \{(x_i,\mathbf{v}_i)\}_{i=1}^\ell \leftarrow \mathcal{A}_2^{O_H^*}(z_O, 1^\ell)\right] \geq \Pr_{H \leftarrow \mathsf{Bias}_{q,1/\lambda^4,\mathbb{F}_q^s}}[\mathsf{SmallSet}_H] \cdot \prod_{i=1}^{\ell}\left(\frac{1}{2p(\lambda)} - \frac{i-1}{2^\lambda}\right).$$

Here we applied the definition of conditional probability and used the fact that since the x_i's are uniformly random, the probability that $x_i \in \bigcup_{j=1}^{i-1}\{x_j\}$ is at most $\frac{i-1}{2^\lambda}$.

As O_H^* has at most $2^{\lambda/2}$ nonzero points, we can hardwire $2^{\lambda/2} \cdot (\lambda + \log|\Sigma|^q) \leq 2^{2\lambda/3}$ bits of advice and simulate O_H^*. By guessing this extra advice along with z_O, we get an algorithm $\mathcal{A}_3$ such that for $\ell = 2^{3\lambda/4}$,

$$\begin{aligned}
&\Pr_{H \leftarrow \mathsf{Bias}_{q,1/\lambda^4,\mathbb{F}_q^s}}\left[\begin{matrix}\forall i \neq j, (x_i,\mathbf{v}_i) \neq (x_j,\mathbf{v}_j) \\ \wedge\, (x_i \| 0^{q-\lambda}, \mathbf{v}_i) \in R_{C_\lambda,H}\end{matrix} : \{(x_i,\mathbf{v}_i)\}_{i=1}^\ell \leftarrow \mathcal{A}_3(1^\ell)\right] \\
&\geq 2^{-(t(\lambda)+2^{2\lambda/3})} \cdot \left(1 - \frac{1}{4p(\lambda)}\right) \cdot \prod_{i=1}^{\ell}\left(\frac{1}{2p(\lambda)} - \frac{i-1}{2^\lambda}\right) \\
&\geq 2^{-\ell} \cdot \prod_{i=1}^{\ell}\left(\frac{1}{2p(\lambda)} - \frac{\ell}{2^\lambda}\right) \geq 2^{-\ell} \cdot \left(\frac{1}{3p(\lambda)}\right)^\ell = \left(\frac{1}{6p(\lambda)}\right)^\ell.
\end{aligned}$$

Applying Lemma 6.6 for $\ell = 2^{3\lambda/4} \leq 2^\lambda$, we have a contradicting upper bound on the success probability of

$$\Pr_{H \leftarrow \mathsf{Bias}_{q,1/\lambda^4,\mathbb{F}_q^s}}\left[\begin{matrix}\forall i \neq j, (x_i,\mathbf{v}_i) \neq (x_j,\mathbf{v}_j) \\ \wedge\, (x_i \| 0^{q-\lambda}, \mathbf{v}_i) \in R_{C_\lambda,H}\end{matrix} : \{(x_i,\mathbf{v}_i)\}_{i=1}^\ell \leftarrow \mathcal{A}_3(1^\ell)\right] \leq \left(1 - \frac{1}{\lambda^4}\right)^{\lambda^5\ell/2} \leq e^{-\lambda\ell/2} \ll \left(\frac{1}{6p(\lambda)}\right)^\ell.$$

□

Having shown that there is a search problem outside of $\mathsf{BQP}/\mathrm{poly}$, we now apply Lemma 4.21. This was essentially established in [LLPY23], but we make minor modifications to deal with quantum queries.

Lemma 7.4. *There is a family of distributions $\{\mathcal{D}_\lambda\}_{\lambda\in\mathbb{N}}$, where $\mathcal{D}_\lambda$ is supported on tuples $(G,\mathcal{O}')$ of functions $G : \{0,1\}^\lambda \to \{0,1\}$ and $\mathcal{O}' : \{0,1\}^{\mathrm{poly}(\lambda)} \to \{0,1\}^{\mathrm{poly}(\lambda)}$, satisfying the following:*

1. *There is an polynomial-time uniform quantum algorithm $\mathcal{A}$ which makes one query to $\mathcal{O}'$ such that for all $\mathcal{O}'$ there exists a family of $\mathrm{poly}(\lambda)$-qubit quantum advice $\{|z_{\mathcal{O}'}\rangle\}_{\mathcal{O}'}$ such that*

$$\Pr_{(G,\mathcal{O}')\leftarrow\mathcal{D}_\lambda}[\forall x \in \{0,1\}^\lambda, \Pr[\mathcal{A}^{\mathcal{O}'}(|z_{\mathcal{O}'}\rangle, x) = G(x)] \geq 1 - \mathrm{negl}(\lambda)] \geq 1 - \mathrm{negl}(\lambda).$$

2. *For all quantum query algorithms algorithm $\mathcal{B}$ that makes $\mathrm{poly}(\lambda)$ queries to $\mathcal{O}'$ and receiving a family of $\mathrm{poly}(\lambda)$-bit classical advice $\{z_{\mathcal{O}'}\}_{\mathcal{O}'}$ depending only on $\mathcal{O}'$,*

$$\Pr_{(G,\mathcal{O}')\leftarrow\mathcal{D}_\lambda, x\leftarrow\{0,1\}^\lambda}[\mathcal{B}^{\mathcal{O}'}(z_{\mathcal{O}'}, x) = G(x)] \leq \frac{3}{5},$$

for all sufficiently large λ.

Proof. Define $\mathcal{D}_\lambda$ as follows: first, sample a random function $G : \{0,1\}^\lambda \to \{0,1\}$ and $H \leftarrow \mathsf{Bias}_{q,1/\lambda^4,\mathbb{F}_q^s}$ and let

$$\mathcal{O}'(x,\mathbf{v}) := \begin{cases} G(x) & \text{if } (x\|0^{q-\lambda},\mathbf{v}) \in R_{C_\lambda,H}, \\ \perp & \text{otherwise.}\end{cases}$$

We begin by showing easiness with quantum advice. Let $(\mathcal{A}', \{|z'_H\rangle\}_H)$ be the algorithm and advice family from Item 1 of Lemma 7.1. We now construct an algorithm $\mathcal{A}$ and family of mixed state advice $\{\rho_{\mathcal{O}'}\}_{\mathcal{O}'}$).[8] We describe a randomized procedure to set $\rho_{\mathcal{O}'}$ given an oracle $\mathcal{O}'$, but in reality, we will set $\rho_{\mathcal{O}'}$ to be the mixed state corresponding to the mixture over outputs of this procedure. Sample (G, H) from the distribution of $\mathcal{D}_\lambda$ conditioned on $\mathcal{O}'$; by construction, the joint distribution of $(G, H, \mathcal{O}')$ sampled in this procedure is identical to $\mathcal{D}_\lambda$. We then set our advice to be $\rho_{\mathcal{O}'} = |z'_H\rangle$. The algorithm $\mathcal{A}$ on input x will run $\mathbf{v} \leftarrow \mathcal{A}'(\rho_{\mathcal{O}'}, x)$, query $(x, \mathbf{v})$ to $\mathcal{O}'$, and output whatever $\mathcal{O}'$ returns. Item 1 of Lemma 7.1 then implies that $\mathcal{A}$ is efficient/uniform and that

$$\Pr_{(G,\mathcal{O}')\leftarrow\mathcal{D}_\lambda}[\forall x \in \{0,1\}^\lambda, \Pr[\mathcal{A}^{\mathcal{O}'}(\rho_{\mathcal{O}'}, x) = G(x)] \geq 1 - \mathrm{negl}(\lambda)] \geq 1 - \mathrm{negl}(\lambda).$$

Now suppose for the sake of contradiction that there was some algorithm $\mathcal{B}$ which made $Q(\lambda) = \mathrm{poly}(\lambda)$ queries and had $t(\lambda) = \mathrm{poly}(\lambda)$-bit classical advice $\{z_{\mathcal{O}'}\}_{\mathcal{O}'}$ such that for infinitely many λ,

$$\Pr_{\substack{(G,\mathcal{O}')\leftarrow\mathcal{D}_\lambda \\ x\leftarrow\{0,1\}^\lambda}}[\mathcal{B}^{\mathcal{O}'}(z_{\mathcal{O}'}, x) = G(x)] > \frac{3}{5}.$$

We know that $\mathcal{O}'$ returns $G(x)$ only if the query $(x, \mathbf{v}) \in R_{C_\lambda, H}$. Thus, by a direct reduction to Lemma 4.21, for a $\frac{1}{4000Q(\lambda)^2}$ fraction of $x \in \{0,1\}^\lambda$, measuring a random query of $\mathcal{B}$ to a randomly sampled oracle $\mathcal{O}' \leftarrow \mathcal{D}_\lambda$ will produce $(x, \mathbf{v}) \in R_{C_\lambda, H}$ with probability at least $\frac{1}{3200Q(\lambda)^2}$.

But now observe that $\mathcal{O}'$ can be simulated by querying G and $\mathcal{O}_H := O[H, \{0,1\}^\lambda \times \Sigma^q]$. Thus, for each function G, we define the following Q-query algorithm $\mathcal{B}'[G]$ and classical advice $\{z'_{\mathcal{O}}[G]\}_{\mathcal{O}}$. First, construct $\mathcal{O}'$ from $(G, \mathcal{O})$ (since $\mathcal{O}$ uniquely determines H) before setting $z'_{\mathcal{O}}[G] := z_{\mathcal{O}'}$. $\mathcal{B}'[G]^{\mathcal{O}}(z'_{\mathcal{O}}[G], x)$ will run $\mathcal{B}^{\mathcal{O}'}(z'_{\mathcal{O}}[G], x)$ where $\mathcal{B}'$ will simulate the oracle $\mathcal{O}'$ using its own oracle $\mathcal{O}$ and the hardwired oracle G and measure a uniformly chosen query of $\mathcal{B}$. As noted earlier, this means that

$$\Pr_{G,H,x}[(x, \mathbf{v}) \in R_{C_\lambda, H} : \mathbf{v} \leftarrow \mathcal{B}'[G]^{\mathcal{O}}(z'_{\mathcal{O}}[G], x)] \geq \frac{1}{3200 \cdot 4000 \cdot Q(\lambda)^4} = \frac{1}{\mathrm{poly}(\lambda)}.$$

By taking G^* which maximizes the above probability,[9] $(\mathcal{B}'[G^*], \{z'_{\mathcal{O}}[G^*]\}_{\mathcal{O}})$ breaks Item 2 of Lemma 7.1. □

A simple diagonalization argument gives us our desired separation (see Section §B for details).

Theorem 7.5. *There is a classical oracle $\mathcal{O}$ such that* $\mathsf{BQP}^{\mathcal{O}}/\mathrm{qpoly} \cap \mathsf{NP}^{\mathcal{O}} \cap \mathsf{coNP}^{\mathcal{O}} \subsetneq \mathsf{BQP}^{\mathcal{O}}/\mathrm{poly}$.[10]

Acknowledgments. We thank Anand Natarajan, Scott Aaronson, Joe Carolan, Ryan Williams, Rohan Goyal, Venkatesan Guruswami, Mary Wootters and Rachel Zhang for patiently answering our many questions. JB is supported by Henry Yuen's AFOSR award FA9550-23-1-0363. VV gratefully acknowledges support from a Simons Investigator Award and a Ford Foundation Chair.

References

[Aar07] Scott Aaronson. The learnability of quantum states. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 463(2088):3089–3114, 2007.

[Aar09] Scott Aaronson. Quantum copy-protection and quantum money. In *2009 24th Annual IEEE Conference on Computational Complexity*, pages 229–242. IEEE, 2009.

[8] This is without loss of generality as a mixed state is a distribution over pure states and so there is always a pure state advice that is at least as good as the mixed state advice.

[9] As noted in [LLPY23], finding G^* does not actually require access to the specific H since $\mathcal{B}'$ can find G^* by itself by using its unbounded computational power to enumerate over all possible G and $\mathcal{O}_H$.

[10] It is not hard to extend this separation to show that $\mathsf{YQP}^{\mathcal{O}} \cap \mathsf{NP}^{\mathcal{O}} \cap \mathsf{coNP}^{\mathcal{O}} \subsetneq \mathsf{BQP}^{\mathcal{O}}/\mathrm{poly}$, where YQP is the class of problems that can be decided by a BQP machine with *untrusted* quantum advice [Aar07, AD14].

[Aar10] Scott Aaronson. Bqp and the polynomial hierarchy. In *Proceedings of the Forty-Second ACM Symposium on Theory of Computing*, STOC '10, page 141–150, New York, NY, USA, 2010. Association for Computing Machinery.

[AD14] Scott Aaronson and Andrew Drucker. A full characterization of quantum advice. *SIAM Journal on Computing*, 43(3):1131–1183, 2014.

[AK07] Scott Aaronson and Greg Kuperberg. Quantum versus classical proofs and advice. In *Twenty-Second Annual IEEE Conference on Computational Complexity (CCC'07)*, pages 115–128. IEEE, 2007.

[AN02] Dorit Aharonov and Tomer Naveh. Quantum np-a survey. *arXiv preprint quant-ph/0210077*, 2002.

[BBBV97] Charles H Bennett, Ethan Bernstein, Gilles Brassard, and Umesh Vazirani. Strengths and weaknesses of quantum computing. *SIAM journal on Computing*, 26(5):1510–1523, 1997.

[BDK24] Shalev Ben-David and Srijita Kundu. Oracle separation of qma and qcma with bounded adaptivity. *arXiv preprint arXiv:2402.00298*, 2024.

[BHNZ25] John Bostanci, Jonas Haferkamp, Chinmay Nirkhe, and Mark Zhandry. Separating qma from qcma with a classical oracle. *arXiv preprint arXiv:2511.09551*, 2025.

[Bla24] Ian F Blake. *Essays on Coding Theory*. Cambridge University Press, 2024.

[BNZ25] John Bostanci, Barak Nehoran, and Mark Zhandry. A general quantum duality for representations of groups with applications to quantum money, lightning, and fire. In *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*, pages 201–212, 2025.

[Bor09] Émile Borel. Les probabilités dénombrables et leurs applications arithmétiques. *Rendiconti del Circolo Matematico di Palermo (1884-1940)*, 27(1):247–271, 1909.

[BV93] Ethan Bernstein and Umesh Vazirani. Quantum complexity theory. In *Proceedings of the twenty-fifth annual ACM symposium on Theory of computing*, pages 11–20, 1993.

[Can17] Francesco Cantelli. Sulla probabilista come limita della frequencza. *Rend. Accad. Lincei*, 26:39, 1917.

[CGLQ20] Kai-Min Chung, Siyao Guo, Qipeng Liu, and Luowen Qian. Tight quantum time-space tradeoffs for function inversion. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 673–684. IEEE, 2020.

[CW02] Cleve and Watrous. Sharp quantum versus classical query complexity separations. *Algorithmica*, 34(4):449–461, 2002.

[DKSS13] Zeev Dvir, Swastik Kopparty, Shubhangi Saraf, and Madhu Sudan. Extensions to the method of multiplicities, with applications to kakeya sets and mergers. *SIAM Journal on Computing*, 42(6):2305–2328, 2013.

[FGH+12] Edward Farhi, David Gosset, Avinatan Hassidim, Andrew Lutomirski, and Peter Shor. Quantum money from knots. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, pages 276–289, 2012.

[FK15] Bill Fefferman and Shelby Kimmel. Quantum vs classical proofs and subset verification. *arXiv preprint arXiv:1510.06750*, 2015.

[GGJL25] Mika Göös, Tom Gur, Siddhartha Jain, and Jiawei Li. Quantum communication advantage in tfnp. In *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*, pages 1465–1475, 2025.

[GS86] S Goldwasser and M Sipser. Private coins versus public coins in interactive proof systems. In *Proceedings of the Eighteenth Annual ACM Symposium on Theory of Computing*, STOC '86, page 59–68, New York, NY, USA, 1986. Association for Computing Machinery.

[GUV09] Venkatesan Guruswami, Christopher Umans, and Salil Vadhan. Unbalanced expanders and randomness extractors from parvaresh–vardy codes. *Journal of the ACM (JACM)*, 56(4):1–34, 2009.

[Kop15] Swastik Kopparty. Some remarks on multiplicity codes. *arXiv preprint arXiv:1505.07547*, 2015.

[KSY14] Swastik Kopparty, Shubhangi Saraf, and Sergey Yekhanin. High-rate codes with sublinear-time decoding. *Journal of the ACM (JACM)*, 61(5):1–20, 2014.

[KTS22] Itay Kalev and Amnon Ta-Shma. Unbalanced expanders from multiplicity codes. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2022)*, pages 12–1. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2022.

[Liu23] Qipeng Liu. Non-uniformity and quantum advice in the quantum random oracle model. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 117–143. Springer, 2023.

[LLPY23] Xingjian Li, Qipeng Liu, Angelos Pelecanos, and Takashi Yamakawa. Classical vs quantum advice and proofs under classically-accessible oracle. *arXiv preprint arXiv:2303.04298*, 2023.

[LMY25] Jiahui Liu, Saachi Mutreja, and Henry Yuen. Qma vs. qcma and pseudorandomness, 2025.

[LS25] Ray Li and Nikhil Shagrithaya. Near-optimal list-recovery of linear code families. *arXiv preprint arXiv:2502.13877*, 2025.

[Lut11] Andrew Lutomirski. Component mixers and a hardness result for counterfeiting quantum money. *arXiv preprint arXiv:1107.0321*, 2011.

[Nie01] Rasmus Refslund Nielsen. *List decoding of linear block codes*. Department of Mathematics, Technical University of Denmark, 2001.

[NN24] Anand Natarajan and Chinmay Nirkhe. A distribution testing oracle separation between qma and qcma. *Quantum*, 8:1377, 2024.

[NY04] Harumichi Nishimura and Tomoyuki Yamakami. Polynomial time quantum computation with advice. *Information Processing Letters*, 90(4):195–204, 2004.

[Reg09] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM (JACM)*, 56(6):1–40, 2009.

[RT97] M Yu Rosenbloom and Michael Anatol'evich Tsfasman. Codes for the m-metric. *Problemy Peredachi Informatsii*, 33(1):55–63, 1997.

[RZVW24] Noga Ron-Zewi, S Venkitesh, and Mary Wootters. Efficient list-decoding of polynomial ideal codes with optimal list size. *arXiv preprint arXiv:2401.14517*, 2024.

[vDHI06] Wim van Dam, Sean Hallgren, and Lawrence Ip. Quantum algorithms for some hidden shift problems. *SIAM Journal on Computing*, 36(3):763–778, 2006.

[WB86] Lloyd R Welch and Elwyn R Berlekamp. Error correction for algebraic block codes, December 30 1986. US Patent 4,633,470.

[YZ24] Takashi Yamakawa and Mark Zhandry. Verifiable quantum advantage without structure. *Journal of the ACM*, 71(3):1–50, 2024.

[Zha24] Mark Zhandry. Toward separating QMA from QCMA with a classical oracle. *arXiv preprint arXiv:2411.01718*, 2024.

[Zha25] Mark Zhandry. Quantum money from abelian group actions. *TheoretiCS*, 4, 2025.

A Duals of Multiplicity Codes

For a field $\mathbb{F}_q$, the multiplicity $\mathsf{mult}(f, \alpha)$ of a polynomial $f \in \mathbb{F}_q[X]$ at a point $\alpha \in \mathbb{F}_q$ is the largest integer m so that $f^{(i)}(\alpha) = 0$ for any non-negative integer $i < m$. The multiplicity Schwartz-Zippel Lemma from [DKSS13] says that a nonzero degree k univariate polynomial can vanish on at most k points, counting multiplicities.

Lemma A.1 ([DKSS13]). *Let $f \in \mathbb{F}_q[X]$ be a nonzero polynomial of degree at most k. Then $\sum_{\alpha \in \mathbb{F}_q} \mathsf{mult}(f, \alpha) \leq k$.*

Fact A.2 (Hasse derivatives, see [Bla24]). *The following properties hold for the Hasse derivative:*

1. *For any polynomial $f(X) \in \mathbb{F}_q[X]$, integer $i \geq 0$, and point $\alpha \in \mathbb{F}_q$, $f^{(i)}(\alpha)$ is the coefficient of X^i in $f(X + \alpha)$.*
2. *(Linearity) For any $f, g \in \mathbb{F}_q[x]$, $\lambda, \mu \in \mathbb{F}_q$, and $i \geq 0$, $(\lambda \cdot f + \mu \cdot g)^{(i)} = \lambda \cdot f^{(i)} + \mu \cdot g^{(i)}$.*
3. *(Product rule) For any $f, g \in \mathbb{F}_q[x]$ and $i \geq 0$, we have $(f \cdot g)^{(i)} = \sum_{k=0}^{i} f^{(k)} \cdot g^{(i-k)}$.*

We first derive a natural analogue of Lagrange interpolation for the setting of Hasse derivatives.

Lemma A.3 (Hermite interpolation). *Let $\mathbb{F}_q$ be a field, $s \geq 1$ be a positive integer, and $\alpha_1, \ldots, \alpha_n$ be distinct points in $\mathbb{F}_q$. For $i \in [n]$ and $0 \leq j \leq s-1$, define $\mu_i(X) := \prod_{i' \neq i} (X - \alpha_{i'})^s$ and $\eta_i(X) := (\mu_i(X))^{-1} \bmod (X - \alpha_i)^s$.*

Then, for all $f(X) \in (\mathbb{F}_q)_{<sn}[X]$, we can write $f(X) = \sum_{i=1}^{n} \sum_{j=0}^{s-1} A_{i,j}(X) f^{(j)}(\alpha_i)$, where

$$A_{i,j}(X) = \mu_i(X)(X - \alpha_i)^j \sum_{t=0}^{s-1-j} \eta_i^{(t)}(\alpha_i)(X - \alpha_i)^t.$$

Proof. We begin by showing that for any $i, i' \in [n]$ and $0 \leq j, j' \leq s-1$, $A_{i,j}^{(j')}(\alpha_{i'}) = 1$ if $(i, j) = (i', j')$ and 0 otherwise. First, if $i' \neq i$, then $A_{i,j}(X + \alpha_{i'}) = X^s \cdot B_{i,j}(X)$ for some polynomial $B_{i,j}(X)$ so $A_{i,j}^{(j')}(\alpha_{i'}) = 0$ for all j, j'. Similarly, since $A_{i,j}(X + \alpha_i) = X^j \cdot C_{i,j}(X)$ for some polynomial $C_{i,j}(X)$, $A_{i,j}^{(j')}(\alpha_i) = 0$ whenever $i = i'$ and $j' < j$. It thus remains to consider $i = i'$ and $j \leq j'$. By the product rule, we know that

$$\begin{aligned} A_{i,j}^{(j')}(\alpha_i) &= \sum_{k=0}^{j'} \mu_i^{(j'-k)}(\alpha_i) \cdot \left[\sum_{t=0}^{s-1-j} \eta_i^{(t)}(\alpha_i)(X - \alpha_i)^{j+t} \right]^{(k)} (\alpha_i) = \sum_{k=j}^{j'} \mu_i^{(j'-k)}(\alpha_i) \cdot \eta_i^{(k-j)}(\alpha_i) \\ &= \sum_{\ell=0}^{j'-j} \mu_i^{(j'-j-\ell)}(\alpha_i) \cdot \eta_i^{(\ell)}(\alpha_i) = (\mu_i \cdot \eta_i)^{(j'-j)}(\alpha_i). \end{aligned}$$

By construction, $(\mu_i \cdot \eta_i)(X) = 1 + h(X) \cdot (X - \alpha_i)^s$ for some polynomial $h(X)$, so $\mu_i(X + \alpha_i) \cdot \eta_i(X + \alpha_i) = 1 + X^s \cdot h(X + \alpha_i)$. As $0 \leq j' - j \leq s-1$, we conclude that $(\mu_i \cdot \eta_i)^{(j'-j)}(\alpha_i)$ equals 1 if $j' = j$ and 0 otherwise.

Now, let $g(X) = \sum_{i=1}^{n} \sum_{j=0}^{s-1} A_{i,j}(X) f^{(j)}(\alpha_i)$. Note that since $\deg A_{i,j}, \deg f \leq sn - 1$, we know that $\deg(g - f) \leq sn - 1$. But for any $i' \in [n]$ and $0 \leq j' \leq s-1$, we have that

$$(g - f)^{(j')}(\alpha_{i'}) = g^{(j')}(\alpha_{i'}) - f^{(j')}(\alpha_{i'}) = \left[\sum_{i=1}^{n} \sum_{j=0}^{s-1} A_{i,j}^{(j')}(\alpha_{i'}) f^{(j)}(\alpha_i) \right] - f^{(j')}(\alpha_{i'}) = f^{(j')}(\alpha_{i'}) - f^{(j')}(\alpha_{i'}) = 0.$$

Thus, by the multiplicity Schwartz-Zippel Lemma (Lemma A.1), $(g - f)(X) = 0$ and so $f(X) = g(X)$. □

Definition A.4 (Generalized multiplicity codes). *For invertible matrices $U_1, \ldots, U_n \in \mathbb{F}_q^{s \times s}$, define the generalized multiplicity (GM) code $\mathsf{GM}_{s,\mathbb{F}_q}(U_1, \ldots, U_n; \alpha_1, \ldots, \alpha_n; k) := \{(U_1 \cdot c_1, \ldots, U_n \cdot c_n) : c \in \mathsf{Mult}_{s,\mathbb{F}_q}(\alpha_1, \ldots, \alpha_n; k)\}$.*

Note that GM codes have distance at least $n - \frac{k-1}{s}$ by Lemma A.1.

Theorem A.5 (Duality of GM codes). *Let $\mathbb{F}_q$ be a field, and $s \geq 1$ be a positive integer, and $\alpha_1, \ldots, \alpha_n$ be distinct points in $\mathbb{F}_q$. Then there exist invertible matrices $U_1, \ldots, U_n \in \mathbb{F}_q^{s \times s}$, so that for any positive integer $k < sn$,*

$$\mathsf{Mult}_{s,\mathbb{F}_q}(\alpha_1, \ldots, \alpha_n; k) = \mathsf{GM}_{s,\mathbb{F}_q}(U_1, \ldots, U_n; \alpha_1, \ldots, \alpha_n; sn - k)^{\perp}.$$

Proof. Consider any pair of polynomials $f(X) \in (\mathbb{F}_q)_{<k}[X]$ and $g(X) \in (\mathbb{F}_q)_{<sn-k}[X]$. Let $h(X) := f(X) \cdot g(X)$, and note that $h(X)$ has degree at most $sn-2$. By Lemma A.3, there exist polynomials $A_{i,j}(X)$ such that

$$h(X) = \sum_{i=1}^{n} \sum_{j=0}^{s-1} A_{i,j}(X) h^{(j)}(\alpha_i).$$

Letting $a_{i,j}$ denote the coefficient of X^{sn-1} in $A_{i,j}(X)$, we see that the coefficient of X^{sn-1} in $h(X)$ is

$$\sum_{i=1}^{n} \sum_{j=0}^{s-1} a_{i,j} h^{(j)}(\alpha_i) = \sum_{i=1}^{n} \sum_{j=0}^{s-1} a_{i,j} \sum_{\ell=0}^{j} f^{(\ell)}(\alpha_i) g^{(j-\ell)}(\alpha_i) = 0,$$

by the product rule and the fact that $\deg h \leq sn-2$. Consider the following anti-triangular matrices:

$$U_i = \begin{bmatrix} a_{i,0} & a_{i,1} & \cdots & a_{i,s-1} \\ a_{i,1} & \iddots & \iddots & 0 \\ \vdots & \iddots & 0 & \vdots \\ a_{i,s-1} & 0 & \cdots & 0 \end{bmatrix} \in \mathbb{F}_q^{s \times s}.$$

If we define $f_i := (f^{(0)}(\alpha_i), \ldots, f^{(s-1)}(\alpha_i)) \in \mathbb{F}_q^s$ and $g_i := (g^{(0)}(\alpha_i), \ldots, g^{(s-1)}(\alpha_i)) \in \mathbb{F}_q^s$, we see that

$$\sum_{i=1}^{n} \langle f_i, U_i \cdot g_i \rangle = \sum_{i=1}^{n} f_i^T \cdot U_i \cdot g_i = \sum_{i=1}^{n} \sum_{j=0}^{s-1} a_{i,j} \sum_{\ell=0}^{j} f^{(\ell)}(\alpha_i) g^{(j-\ell)}(\alpha_i) = 0.$$

We claim that U_i are invertible. To see this, note that by Lemma A.3, for any $i \in [n]$,

$$A_{i,s-1}(X) = \mu_i(X)(X-\alpha_i)^{s-1}\eta_i(\alpha_i) = \eta_i(\alpha_i)(X-\alpha_i)^{s-1} \prod_{i' \neq i} (X-\alpha_{i'})^s,$$

so $a_{i,s-1} = \eta_i(\alpha_i)$. As $\mu_i \cdot \eta_i \equiv 1 \bmod (X-\alpha_i)^s$, $\mu_i(\alpha_i) \cdot \eta_i(\alpha_i) = 1$ and thus $a_{i,s-1} \neq 0$. Consequently,

$$\det(U_i) = a_{i,s-1}^s \cdot \det(J_s) = a_{i,s-1}^s \cdot (-1)^{s(s-1)/2} \neq 0,$$

where J_s is the $s \times s$ reversal/exchange matrix. Thus, we have shown that $\sum_{i=1}^{n} \langle \mathtt{Enc}_C(f)_i, \mathtt{Enc}_{C'}(g)_i \rangle = 0$ for all f and g, where $C := \mathsf{Mult}_{s,\mathbb{F}_q}(\alpha_1, \ldots, \alpha_n; k)$ and $C' := \mathsf{GM}_{s,\mathbb{F}_q}(U_1, \ldots, U_n; \alpha_1, \ldots, \alpha_n; sn-k)$. We conclude that $C = (C')^\perp$ as both C and $(C')^\perp$ are vector spaces of dimension k. □

Corollary A.6 (Theorem 4.19)**.** *For all parameters s, q, and $k < sq$, $(\mathsf{Mult}_{s,\mathbb{F}_q,k})^\perp$ has distance at least $\frac{k+1}{s}$.*

Proof. By Theorem A.5, $\mathsf{dist}((\mathsf{Mult}_{s,\mathbb{F}_q,k})^\perp) = \mathsf{dist}(\mathsf{GM}_{s,\mathbb{F}_q}(U_1, \ldots, U_q; 1, \ldots, q; sq-k)) \geq q - \frac{sq-k-1}{s} = \frac{k+1}{s}$. □

B Diagonalization Arguments

Proof of Theorem 6.8. The proof is nearly identical to that of [BHNZ25], but we include it for completeness. Let $\mathcal{O} : \{0,1\}^* \to \{0,1\}$ be an oracle and let $\mathcal{O}_\lambda$ be the restriction to λ^7-bit inputs, where the lower threshold is λ_0 (per Remark 6.2). We define the unary (promise) language $\mathcal{L}^{\mathcal{O}}$ so that $1^\lambda \in \mathcal{L}^{\mathcal{O}}$ precisely when $\mathcal{O}_\lambda$ is a YES instance and $1^\lambda \notin \mathcal{L}^{\mathcal{O}}$ precisely when $\mathcal{O}_\lambda$ is a NO instance.

We consider only oracles $\mathcal{O}$ such that each restriction to size λ^7-bit inputs encodes either a YES or NO instance, and so the containment $\mathcal{L}^{\mathcal{O}} \in \mathsf{QMA}^{\mathcal{O}}$ follows from Lemma 6.3 (as we can hardcode the values of $\mathcal{L}^{\mathcal{O}}$ on all inputs of length at most λ_0^7). Showing that $\mathcal{L}^{\mathcal{O}} \in \mathsf{AM}^{\mathcal{O}}$ for all $\mathcal{O}$ is simple: Arthur samples $x \leftarrow \{0,1\}^\lambda$ and Merlin responds with any $\mathbf{v} \in \Sigma^q$ such that $(x \| 0^{q-\lambda}, \mathbf{v}) \in R_{C_\lambda, H}$ (which always exists as $H \in \mathsf{Good}$). Arthur accepts iff $\mathcal{O}(x, \mathbf{v}) = 1$. Completeness and soundness follow essentially immediately.

Now we prove the lower bound for QCMA machines. Let $M_1, M_2, \ldots$ be an enumeration of all possible Turing machines. Identify any surjective function $\iota : \mathbb{N} \twoheadrightarrow \mathbb{N}^2$ and define functions $j, a : \mathbb{N} \to \mathcal{N}$ by $(j(\kappa), a(\kappa)) = \iota(\kappa)$.

Define $F : \mathbb{N} \to \mathbb{N}$ so that $F(a)$ is the minimum value such that for all $\lambda \geq F(a)$, any $Q(\lambda) \leq a\lambda^a$ query algorithm with $t(\lambda) \leq a\lambda^a$-length classical witness must misclassify some $\mathcal{O}_\lambda$. By Lemma 6.7, for every integer a, $F(a)$ is well-defined. We identify integers $n_1, n_2, ...$ where the oracles will be defined to be nonzero. Define integers $n_1 := 1 + F(a(1)), n_\kappa := 1 + \max\{F(a(\kappa)), a(\kappa-1)(n_{\kappa-1})^{a(\kappa-1)}\}$. For any $n \in \mathbb{N} \setminus \{n_1, n_2, ...\}$, let $\mathcal{O}$ equal 0 everywhere. For these input lengths, $\mathcal{O}_n$ is trivially a NO instance.

For each $\kappa \in \mathbb{N}$, run $M_{j(\kappa)}$ on input 1^{n_κ} for $a(\kappa)n_\kappa^{a(\kappa)}$ steps and interpret its output as a quantum query circuit $\mathcal{A}_{n_\kappa}$ which accepts a classical witness. For every query that $\mathcal{A}_{n_\kappa}$ makes of length $< n_\kappa$, use the previously generated definitions of the oracle $\mathcal{O}$ to hardcode these answers. For queries $\mathcal{A}_{n_\kappa}$ makes of length $> n_\kappa$, replace the oracle gates with identity circuits. The resulting circuit will be $\mathcal{B}_{n_\kappa}$, which only makes queries of length n_κ. This new algorithm $\mathcal{B}_{n_\kappa}$ can be used to derive an oracle $\mathcal{O}_{n_\kappa}$ by applying Lemma 6.7.

It remains to prove that no $\mathsf{QCMA}^{\mathcal{O}}$ algorithm exists. Assume, for contradiction, that there exists a P-uniform family of oracle circuits $\{\mathcal{A}_\lambda\}$ that solves the code intersection problem with witnesses of length $t(\lambda) = \text{poly}(\lambda)$ and $Q(\lambda) = \text{poly}(\lambda)$ queries. Then, $\{\mathcal{A}_\lambda\}$ appears in the Turing machine enumeration as some M_{j^*} and there exists some a^* such that $t(\lambda), Q(\lambda) \leq a^*\lambda^{a^*}$. As ι is a surjection, there exists a κ^* such that $\iota(\kappa^*) = (j^*, a^*)$. Let $\mathcal{A}_{n_{\kappa^*}}$ be the quantum circuit for inputs of length n_{κ^*}. Since the oracle is defined as being 0 for inputs $\notin \{n_1, n_2, ...\}$ and $n_{\kappa^*+1} > a^* n_{\kappa^*}^{a^*}$, each query gate for inputs of length $> n_{\kappa^*}$ is an identity gate. Thus the circuit $\mathcal{B}_{n_{\kappa^*}}$ has the exact same output as $\mathcal{A}_{n_{\kappa^*}}$ on inputs of size n_{κ^*}. However, using Lemma 6.7, we constructed an oracle $\mathcal{O}_{n_{\kappa^*}}$ that $\mathcal{B}_{n_{\kappa^*}}$ will misclassify. Therefore, $\mathcal{A}_{n_{\kappa^*}}$ will answer incorrectly on input $1^{n_{\kappa^*}}$, completing the proof. □

Proof of Theorem 7.5. Our proof will closely follow [LLPY23]. Suppose that for each λ we generate $(G_\lambda, \mathcal{O}'_\lambda) \leftarrow \mathcal{D}_\lambda$ and define a language $\mathcal{L}^{\mathcal{O}'} := \bigsqcup_{\lambda \in \mathbb{N}} G_\lambda^{-1}(1)$ and an oracle $\mathcal{O}'$ that returns $\mathcal{O}'_{|x|}(x)$ on a query $x \in \{0,1\}^*$. It suffices to show that $\mathcal{L}^{\mathcal{O}'} \in \mathsf{BQP}^{\mathcal{O}'}/\text{qpoly} \cap \mathsf{NP}^{\mathcal{O}'} \cap \mathsf{coNP}^{\mathcal{O}'}$ and $\mathcal{L}^{\mathcal{O}'} \notin \mathsf{BQP}^{\mathcal{O}'}/\text{poly}$ with probability 1.

To see that $\mathcal{L}^{\mathcal{O}'} \in \mathsf{BQP}^{\mathcal{O}'}/\text{qpoly}$ with probability 1, observe that Item 1 of Lemma 7.4 implies that there is a BQP machine $\mathcal{A}^{\mathcal{O}'}$ with polynomial-size quantum advice that decides $\mathcal{L}^{\mathcal{O}'}$ on all x of length λ with probability at least $1 - \frac{1}{\lambda^2}$ for sufficiently large λ. As $\sum_{\lambda=1}^{\infty} \frac{1}{\lambda^2} = \frac{\pi^2}{6} < \infty$, the Borel–Cantelli lemma (Lemma 4.10) implies that $\mathcal{A}^{\mathcal{O}'}$ decides $\mathcal{L}^{\mathcal{O}'}$ for all but finitely many λ with probability 1. By hard-coding all x's where $\mathcal{A}$ and $\mathcal{L}$ disagree, $\mathcal{A}$ can be modified into a $\mathsf{BQP}/\text{qpoly}$ machine $\mathcal{A}'^{\mathcal{O}'}$ that decides $\mathcal{L}^{\mathcal{O}'}$ on all $x \in \{0,1\}^*$ with probability 1.

In addition, for any $x \in \{0,1\}^\lambda$, we can give any $\mathbf{v}$ where $(x \| 0^{q-\lambda}, \mathbf{v}) \in R_{C_\lambda, H}$ to certify $G(x)$. Thus, by a Chernoff/union bound, we know that with probability $1 - \text{negl}(\lambda)$ over $\mathcal{D}_\lambda$, there exists an $\mathsf{NP}/\mathsf{coNP}$ certificate for all $x \in \{0,1\}^\lambda$. We conclude via an identical argument that $\mathcal{L}^{\mathcal{O}'} \in \mathsf{NP}^{\mathcal{O}'} \cap \mathsf{coNP}^{\mathcal{O}'}$ with probability 1.

For a BQP machine $\mathcal{B}$ that takes $\text{poly}(\lambda)$-bit classical advice, we define $S_{\mathcal{B}}(\lambda)$ to be the event over the choice of $(G, \mathcal{O}')$ that there is a $\text{poly}(\lambda)$-bit classical advice family $\{z_{\mathcal{O}'}\}_{\mathcal{O}'}$ such that

$$\Pr[\forall x \in \{0,1\}^\lambda, \mathcal{B}^{\mathcal{O}'}(z_{\mathcal{O}'}, x) = G(x)] \geq \frac{2}{3}.$$

By Item 2 of Lemma 7.4, there exists $\lambda_0 \in \mathbb{N}$ such that for all BQP machines $\mathcal{B}$, $\Pr_{\mathcal{D}_\lambda}[S_{\mathcal{B}}(\lambda)] \leq \frac{9}{10}$ for all $\lambda \geq \lambda_0$.

We will consider a sequence of input lengths $\lambda_1, \lambda_2, ...$ defined by $\lambda_i := T(\lambda_{i-1}) + 1$, where $T(\lambda)$ is the running time of $\mathcal{B}$ on input of length λ. This means that when $\mathcal{B}$'s input length is λ_{i-1}, it cannot query the oracle on input lengths $\geq \lambda_i$, so it must be the case that

$$\Pr[S_{\mathcal{B}}(\lambda_i) | S_{\mathcal{B}}(\lambda_0) \wedge ... \wedge S_{\mathcal{B}}(\lambda_{i-1})] = \Pr[S_{\mathcal{B}}(\lambda_i)]$$

$$\implies \Pr[S_{\mathcal{B}}(1) \wedge S_{\mathcal{B}}(2) \wedge ...] \leq \Pr\left[\bigwedge_{i=0}^{\infty} S_{\mathcal{B}}(\lambda_i)\right] = \prod_{i=0}^{\infty} \Pr[S_{\mathcal{B}}(\lambda_i) | S_{\mathcal{B}}(\lambda_0) \wedge ... \wedge S_{\mathcal{B}}(\lambda_{i-1})] \leq \prod_{i=0}^{\infty} \frac{9}{10} = 0.$$

But there are countably many BQP machines, so $\Pr[\exists \mathcal{B} : S_{\mathcal{B}}(1) \wedge S_{\mathcal{B}}(2) \wedge ...] = 0$. We conclude that $\mathcal{L}^{\mathcal{O}'} \notin \mathsf{BQP}^{\mathcal{O}'}/\text{poly}$ with probability 1 over the choice of $(G, \mathcal{O}')$, as desired. □