The Compressed Oracle Is A Worthy (Multiplicative) Adversary

Stacey Jeffery
QuSoft, Amsterdam, The Netherlands
CWI, Amsterdam, The Netherlands
University of Amsterdam, The Netherlands

Sebastian Zur[1]
IRIF, Paris, France
CNRS, Paris, France

Abstract

The compressed oracle technique, introduced in the context of quantum cryptanalysis, is the latest method for proving quantum query lower bounds, and has had an impressive number of applications since its introduction, due in part to the ease of importing classical lower bound intuition into the quantum setting via this method. Previously, the main quantum query lower bound methods were the polynomial method, the adversary method, and the multiplicative adversary method, and their relative powers were well understood. In this work, we situate the compressed oracle technique within this established landscape, by showing that it is a special case of the multiplicative adversary method. To accomplish this, we introduce a simplified restriction of the multiplicative adversary method, the MLADV method, that remains powerful enough to capture the polynomial method and exhibit a strong direct product theorem, but is much simpler to reason about. We show that the compressed oracle technique is also captured by the MLADV method. This might make the MLADV method a promising direction in the current quest to extend the compressed oracle technique to non-product distributions.

2012 ACM Subject Classification Theory of computation → Quantum complexity theory

Keywords and phrases Quantum query complexity, compressed oracle, adversary method

Digital Object Identifier 10.4230/LIPIcs.ICALP.2026.118

Category Track A: Algorithms, Complexity and Games

Related Version *Full Version*: `https://arxiv.org/pdf/2509.07876` [22]

Funding *Stacey Jeffery*: This work is supported by ERC STG grant 101040624-ASC-Q and NWO Klein project number OCENW.Klein.061. SJ is a CIFAR Fellow in the Quantum Information Science Program.

1 Introduction

Proving quantum query lower bounds is essential to understanding the limitations of quantum computers. In the *bounded-error quantum query model*, an algorithm for a problem F receives its input – typically a string in $[M]^N$ for some integers M and N – encoded as a function $f : [N] \to [M]$, accessible only through queries. The algorithm may alternate such queries with arbitrary quantum operations, and it must produce the correct output on every input with probability at least $2/3$. The *bounded-error quantum query complexity* of F, denoted $Q(\mathsf{F})$, is the minimum number of queries needed by any such algorithm. Allowing some small probability of error makes this a practical model of computation, and lower bounds on the query complexity of an algorithm are also lower bounds on the total number of steps the algorithm must make.

[1] The majority of this work was conducted while SZ was affiliated with CWI & QuSoft, the Netherlands.

53rd International Colloquium on Automata, Languages, and Programming (ICALP 2026).
Editors: Sayan Bhattacharya, Danupon Nanongkai, Michael Benedikt, and Gabriele Puppis;
Article No. 118; pp. 118:1–118:23


Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

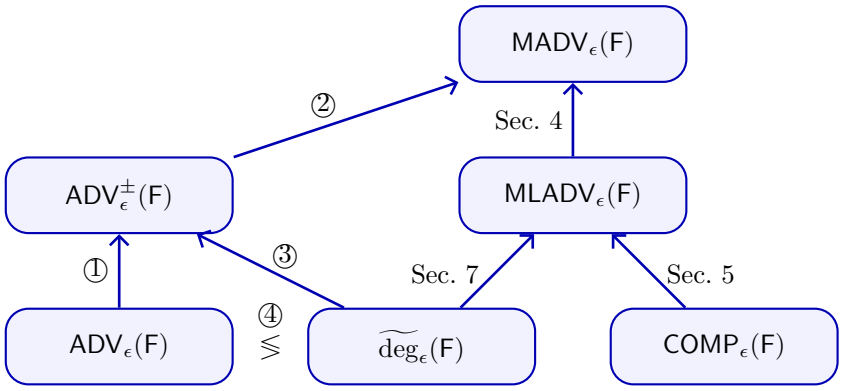

Figure 1 The relationships between the various methods to obtain quantum query lower bounds, expanding on a similar figure in [27]. An arrow from method A to method B implies that for any lower bound that can be proven with A, we can explicitly construct a lower bound with B (i.e., B is stronger than A). ① [21]; ② [6]; ③ [10] only holds in the bounded-error regime; ④ The original additive and the polynomial methods are incomparable [33, 4]. Technically, there are two slightly different definitions of $\mathsf{MLADV}_\epsilon(\mathsf{F})$ defined in this work: a simpler to state one as in Theorem 23, and a stronger one in Theorem 33. This mirrors the situation with $\mathsf{MADV}_\epsilon(\mathsf{F})$, which can denote the slightly weaker bound from [31], or the stronger variant from [24] that is slightly more complicated to state, though no more complicated to apply. In this figure, we mean the stronger version of both.

The first technique for proving quantum query lower bounds was the *polynomial method* [9], which showed that the acceptance probability of a quantum algorithm can be represented by a low-degree polynomial. In this method, one lower bounds the quantum query complexity of F by lower bounding its *approximate degree* $\widetilde{\deg}(\mathsf{F})$, by proving a lower bound on the degree of any polynomial with certain properties that must be satisfied by any successful algorithm. Later, the *adversary method* was introduced in [3] and generalized to its full version in [21]. Letting $\mathsf{ADV}^{\pm}(\mathsf{F})$ denote the best possible lower bound on the quantum query complexity of F that one can prove using the adversary method, it was later shown that this quantity is equal, up to constants, to $Q(\mathsf{F})$, making this a very powerful method. In contrast, there are problems for which the polynomial method is not able to prove tight lower bounds, since $\widetilde{\deg}(\mathsf{F}) = o(Q(\mathsf{F}))$ [4]. However, the polynomial method has an advantage over the adversary method. While the adversary method is only able to prove non-trivial lower bounds on *bounded-error* quantum query complexity, the polynomial method can be used to prove lower bounds on $Q_\epsilon(\mathsf{F})$, the minimum number of queries needed by any quantum algorithm to compute F with success probability at least $1-\epsilon$, even when $1-\epsilon = o(1)$. This is particularly useful in cryptographic settings, as we discuss shortly.

A later more powerful variant of the adversary method is the *multiplicative adversary method* [31], which improves on the adversary method by allowing for lower bounds on $Q_\epsilon(\mathsf{F})$ even when the success probability $1-\epsilon$ is very small. This method is at least as powerful as both the adversary method and the polynomial method, but it has few applications simply because it is very difficult to apply. In both lower bound techniques and algorithmic techniques, there is often a tradeoff between the power of a technique, and its ease of application, and an important pursuit is to find techniques with just the right balance of power and ease of use.

A relative newcomer to the landscape of quantum query lower bound techniques is the *compressed oracle technique*. This was introduced in [32], and distilled into a formal framework in [14]. The compressed oracle framework was introduced in the context of post-quantum

cryptanalysis, in order to lower bound the work needed by a quantum *adversary* that interacts with a *quantum random oracle* – a uniform random function $f : X \to Y$ that the adversary can query in superposition. As such an adversary's goal is generally something nefarious, it is critical to show the impossibility of efficient adversaries that achieve their goal, even with success probabilities below a constant. This technique has received widespread use since its introduction, and seems to have a particularly nice balance of power and ease-of-use. There is a nice intuition behind the technique that makes it particularly well suited for adapting classical intuitions about the hardness of a problem to the quantum world, but it has also been powerful enough to prove a number of new results. Still, its use has remained mostly restricted to the setting of uniform functions, and it has proven resistant to relatively minor modifications, such as a generalisation to non-product distributions.

While the relationship between the other mentioned methods is well established, prior to this work, it was unknown where the compressed oracle method fit into the picture.

Contributions

In this work, we show that the compressed oracle technique can be viewed as a special case of the multiplicative adversary method, fitting it into the existing landscape of quantum query lower bound techniques. To do this, we introduce a simplification of the multiplicative adversary method called the *multiplicative ladder adversary method* (MLADV_ϵ, Section 4), and show that the compressed oracle technique COMP_ϵ is actually a special case of this restricted method (Section 5). This restriction of the multiplicative adversary introduces intuitive structure that makes it, in principle, easier to apply, but we show that it is still powerful enough to capture the polynomial method (Section 7). In fact, to the best of our knowledge, virtually every use of the multiplicative adversary to date is an instance of a multiplicative ladder adversary. As further evidence of its natural structure, we show (Section 6) that MLADV_ϵ exhibits a strong direct product theorem. These relationships (and others) are summarized in Figure 1.

We now give a more detailed survey of quantum query lower bound techniques, and discussion of our results.

1.1 Adversary methods

The original adversary method ADV_ϵ for proving quantum query lower bounds was first introduced in [3]. Applying this method reduces to mostly combinatorial arguments, which makes it very convenient to use, as shown by its many applications [8, 18, 13, 17]. However, this method does have some technical limitations, one of which is the *certificate complexity barrier* [33], which shows that there are problems for which this method cannot be tight. This limitation is addressed by the strictly stronger negative-weights adversary method $\mathsf{ADV}^{\pm}_\epsilon$ by [21], now usually just referred to as *the adversary method*. This method is capable of proving tight lower bounds on the quantum query complexity of any F in the bounded-error regime [28], but this power comes at the cost of making it more complicated to apply, as its greater abstraction removes the primarily combinatorial reasoning suggested by the constraints of the original adversary method. This means that even for very symmetric problems such as the *collision problem* [1], it is highly difficult to come up with a non-trivial lower bound using the adversary method, and the only known construction relies on studying the symmetries of the problem via representation theory [11]. Lower bounds on $Q_\epsilon(\mathsf{F})$ proven using the adversary method are proportional to $1 - \epsilon$, the algorithm's success probability, making them negligible for exponentially small success probabilities. It is therefore only suitable for proving lower bounds in the bounded-error regime.

The latest and most powerful iteration in adversary methods, and the one most central to this work, is the multiplicative adversary method MADV_ϵ formalized in [31, 24], as a generalisation of an ad-hoc technique proposed in [7, 5]. This method is shown to be strictly stronger than the adversary method [6]. Since the adversary method is already tight in the bounded-error regime, this generalisation is particularly relevant in the low success probability regime, where it works even for exponentially small probabilities of success. This is a necessary condition for the method to exhibit a *strong direct product theorem* (SDPT), which intuitively states that to solve k independent instances of a function, one needs $\Omega(k)$ times as many queries to achieve even an exponentially small (in k) probability of success. It was already shown in [31] that the multiplicative adversary method satisfies a SDPT, which allowed [24] to prove a SDPT for quantum query complexity. The multiplicative adversary's applicability to the small success probability regime has also proven useful in proving quantum time-space tradeoff lower bounds [7]. However, just as the (negative-weights) adversary method is more complicated to apply than the original adversary method, the powerful multiplicative adversary method is even more complicated and, as a result, still has relatively few applications.

1.2 Polynomial method

Another technique for proving quantum query lower bounds is the polynomial method [9], which predates the adversary method. The method relies on the principle that for any T-query quantum algorithm, its acceptance probability can be expressed as a multivariate polynomial of degree $2T$ in the input variables. In any algorithm that computes F with error ϵ, this polynomial gives an ϵ-approximation to F, and so its degree, $2T$, is at least $\widetilde{\deg}_\epsilon(\mathsf{F})$, the minimum degree of any polynomial that ϵ-approximates F. This allows one to prove lower bounds on $Q_\epsilon(\mathsf{F})$ using results about polynomials. For example, the fact that a polynomial that changes value many times must have high degree implies a lower bound on problems like parity, that change value many times. Using much more involved reasoning, the polynomial method was used to give a tight lower bound on the collision problem [1], whereas an adversary lower bound for this problem was only constructed much later [11].

While the polynomial method is incomparable to the original adversary method [33, 4], the (negative-weights) adversary method subsumes it in the bounded-error regime; a reduction recently made constructive by [10]. The polynomial method does, however, work for small success probabilities, which makes it possible to prove SDPTs [23, 30]. Furthermore, as shown by [27], it can be reduced to the multiplicative adversary method.

1.3 Compressed oracle technique

For cryptographic security proofs, lower bounds on bounded-error quantum query complexity make little sense. Ruling out adversaries that succeed with a high probability of success (at least 2/3) is not enough, and it is necessary to rule out adversaries with very small success probabilities as well. Moreover, worst-case query complexity, as captured by $Q_\epsilon(\mathsf{F})$, is not the relevant quantity. Imagine an adversary whose task F is to break some cryptosystem using its public key as input. A lower bound on $Q_\epsilon(\mathsf{F})$ only proves that there is *some* key on which the adversary requires many queries; it says nothing about the adversary's resource requirements on a *random* key. This is instead captured by *average-case* quantum query complexity, which is defined with respect to some distribution of interest (often the uniform distribution).

The compressed oracle technique [32], introduced in the context of cryptographic security proofs, does precisely this, as it yields an upper bound on the probability of success for any quantum algorithm interacting with a random oracle, giving an average-case lower bound[2] that holds even for exponentially small probabilities of success. Moreover, its analysis works via mostly combinatorial arguments that look quite similar to the types of reasoning one would use to prove classical lower bounds, which makes it straightforward to apply and has quickly resulted in many results [25, 15, 26, 14, 19, 16]. It also satisfies a SDPT, and has even been used to prove quantum time-space tradeoffs [20]. The limitation of this technique, however, is that it is not known how to apply it on input distributions where the values $f(x)$ for different x are not independent [15, 20]. This limitation makes it difficult to prove worst-case lower bounds, as the hardest input distributions often have some global structure. It also rules out applications involving certain interesting cryptographic primitives such as random permutations. For that specific case, an ad-hoc workaround has recently been devised by [2], extending the indifferentiability of the Sponge construction [12] to the quantum setting. From the standpoint of quantum query lower bounds, however, this remedy is no longer tight.

1.4 COMP_ϵ vs. other techniques

All methods discussed above operate by tracking some progress measure that must change significantly over the course of the algorithm, but can only change a small amount using a single query. For the polynomial method, this is the degree of the polynomials representing the amplitudes of the algorithm's states. For the compressed oracle and the adversary methods, the measure of progress is somehow measuring the amount of entanglement between the algorithm and the input, when instantiated as a coherent superposition. Since the adversary and compressed oracle techniques have different drawbacks that do not seem to exist in the other, it is interesting to see what the explicit relationship between these techniques is. This could aid in the ongoing search for a fusion of both techniques: a compressed oracle technique that can be applied to input distributions where each $f(x)$ is not necessarily assigned independently. On the cryptographic side, this could lead to (better) quantum security proofs for schemes using random permutations, such as the sponge construction [12]. On the quantum query lower bounds side, this might result in a technique that marries the power of the multiplicative adversary method – which works for all input distributions – with the intuitive combinatorial reasoning of the compressed oracle technique. Currently, the most promising result towards this "holy grail" has been a representation theory approach by [29] that allows for tackling the problem of inverting a random permutation.

In this work, we demonstrate that a generalised compressed oracle technique – one that accommodates distributions beyond random functions and permutations – must fall somewhere between the compressed oracle technique and the multiplicative adversary method. We explicitly show this by proving that the compressed oracle technique reduces to the multiplicative adversary method. We achieve this by defining a weaker version of the multiplicative adversary method, the multiplicative ladder adversary MLADV_ϵ. An adversary lower bound (multiplicative or standard) is proven by exhibiting an *adversary matrix* that satisfies certain properties. In the MLADV_ϵ technique, we restrict adversary matrices to

[2] Adversary methods also work by giving an average-case lower bound with respect to some distribution of inputs, which then implies a lower bound on the worst-case complexity. However, as the goal in using these is usually a worst-case lower bound, generally a deliberately hard distribution is chosen, rather than a uniform one.

those whose eigenvalues are increasing powers of some constant larger than 1, and whose eigenspaces form a "ladder" in the sense that a query can move the state up or down at most one eigenspace. This ladder structure makes reasoning about an algorithm's progress much more tractable.

The MLADV_ϵ method still satisfies a strong direct product theorem (SDPT, see Section 6) and remains more powerful than the compressed oracle technique. Additionally, we show that this new version also still encompasses the polynomial method (see Section 7). These results are summarized in Figure 1. We hope that this new intermediate technique will aid in the search for an extended compressed oracle technique, as we show that it incorporates the approach from [29] to random permutations as a special case.

2 Preliminaries

2.1 Linear algebra

In this work we consider finite-dimensional complex inner product spaces $\mathcal{H} = \mathbb{C}^d$ for some dimension d. We use standard bra-ket notation for column and row vectors in $\mathbb{C}^d$. We consider all bra-ket vectors to be normalised unless specified otherwise. For a finite set S, we let

$$\mathbb{C}^S = \mathbb{C}[S] = \mathrm{span}\{|s\rangle : s \in S\},$$

using whichever notation is most convenient given the complexity of writing S. For any two Hermitian operators A, B, we write $A \succeq B$ if their difference $A - B$ is positive semidefinite.

▶ **Definition 1** (Spectral norm). *Let $A \in \mathbb{C}^{d\times d}$ be a matrix. Then the* spectral norm *(also known as the operator norm) of A is*

$$\|A\| := \sup_{|v\rangle\in\mathbb{C}^d} \|A|v\rangle\|,$$

where $\|A|v\rangle\|$ is the standard vector ℓ_2-norm.

We will make use of the following standard result.

▶ **Lemma 2.** *For any linear operator A, the spectral norm of A satisfies*

$$\|A\| \le \sqrt{\|A\|_1 \|A\|_\infty}.$$

2.2 Quantum query complexity

In the quantum query model, we are generally interested in computing a function $\mathsf{F} : \mathsf{Func} \to \Sigma$ on an input $f \in \mathsf{Func}$. We consider the case where Func is a subset of Y^X, so each f can itself also be viewed as a function from X to Y. For example, if $Y = \{0, 1\}$ and $X = [n] := \{1, \ldots, n\}$, then $f \in \mathsf{Func}$ is an n-bit string (which might have a promise defined by the subset Func). In this work, we usually restrict ourselves to X being any finite set of size N and consider Y to be the finite set $[M-1]_0 := \{0, \ldots, M-1\}$.

The memory of our quantum algorithm $\mathcal{A}$, tasked with computing F on an input f, is described without loss of generality by the registers $\mathcal{W}$, $\mathcal{X}$, and $\mathcal{Y}$. Here, the input oracle acts on $\mathcal{X} \times \mathcal{Y}$ (as detailed below), while $\mathcal{W}$ represents an additional workspace. The input function $f \in \mathsf{Func}$ can be accessed by $\mathcal{A}$ via an oracle, defined as follows:

▶ **Definition 3** (Oracle). *Fix a finite set X of size N and let $Y = [M-1]_0$. An* oracle $\mathcal{O}_f$, *encoding the input function $f \in \mathsf{Func}$, is a unitary transformation that acts on*

$$\mathrm{span}\{|x\rangle_{\mathcal{X}}|y\rangle_{\mathcal{Y}} : x \in X, y \in Y\},$$

with its action on the basis state $|x\rangle_{\mathcal{X}}|y\rangle_{\mathcal{Y}}$ defined as

$$\mathcal{O}_f|x\rangle_{\mathcal{X}}|y\rangle_{\mathcal{Y}} = |x\rangle_{\mathcal{X}}|(y + f(x)) \bmod M\rangle_{\mathcal{Y}}.$$

The input f is typically drawn from some (hard) input distribution δ over Func, denoted $f \sim \delta$. Consequently, $\mathcal{O}_f$ is a random variable. In adversary methods and the compressed oracle technique, this randomness is avoided by introducing an additional *input* register $\mathcal{I}$, which stores a superposition of function tables representing the input f. In quantum information theory, this is known as *purification*. If $f \sim \delta$, the register $\mathcal{I}$ will be initialised as

$$|\delta\rangle = \sum_{f \in Y^X} \sqrt{\delta(f)}|f\rangle_{\mathcal{I}}.$$

Here, $|\delta\rangle$ represents the initial state of the input register. It is important to note that this should not be confused with the initial state of the algorithm, which is the all-zero state. This purification of the input leads to the following purified oracle:

▶ **Definition 4** (Purified Oracle). *Fix a finite set X of size N and let $Y = [M-1]_0$. A* purified oracle $\mathcal{O}$ *is a unitary transformation that acts on*

$$\mathrm{span}\{|x\rangle_{\mathcal{X}}|y\rangle_{\mathcal{Y}}|f\rangle_{\mathcal{I}} : x \in X, y \in Y, f \in Y^X\},$$

with its action on the basis state $|x\rangle_{\mathcal{X}}|y\rangle_{\mathcal{Y}}|f\rangle_{\mathcal{I}}$ defined as

$$\mathcal{O}|x\rangle_{\mathcal{X}}|y\rangle_{\mathcal{Y}}|f\rangle_{\mathcal{I}} = |x\rangle_{\mathcal{X}}|(y + f(x)) \bmod M\rangle_{\mathcal{Y}}|f\rangle_{\mathcal{I}}.$$

From the perspective of the algorithm, it is indistinguishable whether it interacts with the random variable $\mathcal{O}_f$ or the purified oracle $\mathcal{O}$ with input register initialised to $|\delta\rangle$. The relationship between the two is captured by the following expression:

$$\mathcal{O} = \sum_{f \in Y^X} \mathcal{O}_f \otimes |f\rangle\langle f|_{\mathcal{I}}.$$

It is equivalent, and in this work more convenient, to encode the query into the phase by viewing the $\mathcal{Y}$ register in the Fourier basis $\{|\hat{y}\rangle\}_{y \in Y}$ instead of the computational basis $\{|y\rangle\}_{y \in Y}$.

▶ **Definition 5** (Fourier basis). *Let $Y = [M-1]_0$ and let $\{|y\rangle\}_{y \in Y}$ be the computational basis for $\mathcal{Y} = \mathbb{C}^M$. Then $\{|\hat{y}\rangle\}_{y \in Y}$ is the* Fourier basis *of $\mathcal{Y}$, where each $|\hat{y}\rangle$ is defined as*

$$|\hat{y}\rangle = \frac{1}{\sqrt{M}} \sum_{z \in Y} e^{\frac{2\pi\iota}{M} yz}|z\rangle.$$

Here ι denotes the imaginary unit to prevent ambiguity with the variable i. The unitary map $|y\rangle \mapsto |\hat{y}\rangle$ is also known as the Quantum Fourier Transform over the integers mod M, *which we denote* QFT_M.

In this Fourier basis, the oracle from Definition 4 acts on any basis state $|x\rangle_{\mathcal{X}}|\hat{y}\rangle_{\mathcal{Y}}|f\rangle_{\mathcal{I}}$ as

$$\mathcal{O}|x\rangle_{\mathcal{X}}|\hat{y}\rangle_{\mathcal{Y}}|f\rangle_{\mathcal{I}} = e^{\frac{2\pi\iota}{M} yf(x)}|x\rangle_{\mathcal{X}}|\hat{y}\rangle_{\mathcal{Y}}|f\rangle_{\mathcal{I}}.$$

Additionally, it will often be convenient to decompose the oracle $\mathcal{O}$ into diagonal unitary matrices $\mathcal{O}_{x,y}$ given by

$$\mathcal{O} = \sum_{x\in X, y\in Y} |x\rangle\langle x|_{\mathcal{X}} \otimes |\hat{y}\rangle\langle\hat{y}|_{\mathcal{Y}} \otimes \mathcal{O}_{x,y}, \tag{1}$$

where each $\mathcal{O}_{x,y}$ acts on the basis state $|f\rangle_{\mathcal{I}}$ as

$$\mathcal{O}_{x,y}|f\rangle_{\mathcal{I}} = e^{\frac{2\pi\iota}{M} y\cdot f(x)}|f\rangle_{\mathcal{I}}.$$

▶ **Definition 6** (T-Query Quantum Algorithm). *Fix a set X of size N and let $Y = [M-1]_0$. A T-*query quantum algorithm *$\mathcal{A}$ on Y^X is a sequence of unitaries $U_0, \ldots, U_T$ on*

$$\mathrm{span}\{|w\rangle_{\mathcal{W}}|x\rangle_{\mathcal{X}}|y\rangle_{\mathcal{Y}} : w \in W, x \in X, y \in Y\},$$

for some finite set W. For a fixed algorithm $\mathcal{A}$ and a fixed input distribution δ, let

$$|\delta\rangle = \sum_{f\in Y^X} \sqrt{\delta(f)}|f\rangle_{\mathcal{I}},$$

and let

$$|\psi_t(\mathcal{A},\delta)\rangle = U_t\mathcal{O}U_{t-1}\mathcal{O}\ldots\mathcal{O}U_0|0\rangle_{\mathcal{W}\mathcal{X}\mathcal{Y}}|\delta\rangle_{\mathcal{I}}$$

denote the state of the algorithm before the $(t+1)$-th query is made, and let

$$\rho^t_{\mathcal{I}}(\mathcal{A},\delta) = \mathrm{Tr}_{\mathcal{W}\mathcal{X}\mathcal{Y}}\left[|\psi_t(\mathcal{A},\delta)\rangle\langle\psi_t(\mathcal{A},\delta)|\right]$$

denote the reduced state of the input register, which we call the input register states for $\mathcal{A}$ and $|\delta\rangle$. *When $\mathcal{A}$ and $|\delta\rangle$ are clear from context, we will omit the $(\mathcal{A},\delta)$ notation.*

In the definition of $|\psi_t(\mathcal{A},\delta)\rangle$, both the queries $\mathcal{O}$ and the unitaries $U_1,\ldots,U_t$ act on a larger Hilbert space than originally defined, but each operator is implicitly understood to act tensored with the identity operator on any unaffected registers.

In this work, we compare various techniques designed to lower bound the quantum query complexity of a problem F:

▶ **Definition 7** (ϵ-error Quantum Query Complexity). *Fix $\mathsf{F} : \mathsf{Func} \to \Sigma$. Then the* ϵ-error quantum query complexity *of F, denoted by $Q_\epsilon(\mathsf{F})$, is the minimum number of queries needed by any quantum query algorithm $\mathcal{A}$ to successfully output $\mathsf{F}(f)$ for every input $f \in \mathsf{Func}$ with success probability at least $1-\epsilon$.*

3 The frameworks

In this section, we introduce the two main lower bound frameworks that will be compared throughout this work: the multiplicative adversary method and the compressed oracle technique. The other lower bound method discussed in this paper, the polynomial method, is not needed until Section 7, and we define it there.

3.1 The multiplicative adversary method

The general idea behind the adversary methods is that any algorithm for F, run on a superposition of different inputs $|\delta\rangle$ with different values of F, must entangle the algorithm's workspace $\mathcal{W}\mathcal{X}\mathcal{Y}$ (which must eventually contain the answer) with the input register $\mathcal{I}$, resulting in the reduced density matrix on $\mathcal{I}$, which is initially the pure state $\rho^0_{\mathcal{I}}(\mathcal{A},\delta) = |\delta\rangle\langle\delta|$, becoming some mixed state $\rho^T_{\mathcal{I}}(\mathcal{A},\delta)$.

This idea was already present in the original *quantum adversary method* [3], which was later generalised to the stronger *negative-weights adversary method* [21] (now often called the adversary method), which is tight in the bounded-error regime, i.e. $\epsilon \leq 1/3$. We will be interested in the even more powerful *multiplicative adversary method*, first formalised in [31] and further developed in [6, 24, 27]. We now describe this method.

▶ **Definition 8** (Multiplicative Adversary Matrix). *Fix* $\mathsf{F} : \mathsf{Func} \to \Sigma$. *A* multiplicative adversary matrix *for problem* F *is a positive definite matrix* $\Gamma \in \mathbb{C}^{\mathsf{Func}\times\mathsf{Func}}$ *with smallest eigenvalue 1.*

Any multiplicative adversary matrix gives rise to a *progress measure*, which is a way of quantifying how much progress a quantum algorithm $\mathcal{A}$ has made after t queries towards solving a particular problem F.

▶ **Definition 9** (Progress). *Fix a problem* $\mathsf{F} : \mathsf{Func} \to \Sigma$, *and input distribution* δ *supported on* Func. *Fix a multiplicative adversary matrix* Γ *for* F, *as in Definition 8, with eigenstate* $|\delta\rangle$ *and a* T*-query quantum algorithm* $\mathcal{A}$, *as in Definition 6. Let* $\rho^t_{\mathcal{I}}(\mathcal{A}, \delta)$ *be the input register states for* $\mathcal{A}$ *and input distribution* δ *before the* $(t+1)$*-th query is made. The associated* progress measure *for* $t \in [T]_0$ *is defined as*

$$W^t(\Gamma, \mathcal{A}) \coloneqq \mathrm{Tr}[\Gamma \rho^t_{\mathcal{I}}(\mathcal{A}, \delta)].$$

Theorem 10 quantifies in what way we can think of $W^t(\Gamma, \mathcal{A})$ as a "progress measure." After 0 queries, we have made no progress, which is indicated by $W^0(\Gamma, \mathcal{A}) = 1$ (Item 1). After T queries, if we want to claim that the algorithm actually solves F with probability $1-\epsilon$, then it must be the case that the progress $W^T(\Gamma, \mathcal{A})$ has increased sufficiently above 1 (Item 3). Item 2 bounds the amount of progress that can be made in a single query.

▶ **Theorem 10** ([31, 6]). *Fix a problem* $\mathsf{F} : \mathsf{Func} \to \Sigma$, *an input distribution* δ *on* Func, *and a multiplicative adversary matrix* Γ *for* F *with 1-eigenstate* $|\delta\rangle$. *Let* λ *be a real number with* $1 < \lambda \leq \|\Gamma\|$. *Let* Λ_{bad} *be the projector onto the eigenspaces of* Γ *corresponding to eigenvalues smaller than* λ *and let* $\eta \leq 1-\epsilon$ *be a positive constant such that* $\|F_z \Lambda_{\mathsf{bad}}\|^2 \leq \eta$ *for every* $z \in \Sigma$, *where* $F_z = \sum_{\substack{f\in\mathsf{Func}:\\ \mathsf{F}(f)=z}} |f\rangle\langle f|$. *Then:*

1. *For any quantum algorithm* $\mathcal{A}$, $W^0(\Gamma, \mathcal{A}) = 1$.
2. *For any* T*-query quantum algorithm* $\mathcal{A}$, *and* $t \in [T-1]_0$,

$$\frac{W^{t+1}(\Gamma, \mathcal{A})}{W^t(\Gamma, \mathcal{A})} \leq \max_{x\in X, y\in Y} \left\| \mathcal{O}^\dagger_{x,y} \Gamma^{1/2} \mathcal{O}_{x,y} \Gamma^{-1/2} \right\|^2.$$

3. *For any* T*-query quantum algorithm* $\mathcal{A}$ *that solves* F *on input* $|\delta\rangle$ *with success probability at least* $1-\epsilon$, $W^T(\Gamma, \mathcal{A}) \geq 1 + (\lambda - 1)\left(\sqrt{1-\epsilon} - \sqrt{\eta}\right)^2$.

▶ **Corollary 11.** *For any* η *that satisfies the constraints of Theorem 10,* $\epsilon \in (0, 1-\eta)$, *problem* $\mathsf{F} : \mathsf{Func} \to \Sigma$, *and input distribution* δ *on* Func,

$$Q_\epsilon(\mathsf{F}) \geq \max_{\Gamma,\lambda} \frac{\log\left(1 + (\lambda-1)\left(\sqrt{1-\epsilon} - \sqrt{\eta}\right)^2\right)}{\log\left(\max_{x\in X, y\in Y} \left\| \mathcal{O}^\dagger_{x,y} \Gamma^{1/2} \mathcal{O}_{x,y} \Gamma^{-1/2} \right\|^2\right)},$$

where Γ *ranges over all multiplicative adversary matrices for* F *with* 1*-eigenstate* $|\delta\rangle$ *(see Definition 8) and* λ *ranges over* $[1, \|\Gamma\|]$.

3.2 Dealing with search problems

By Definition 7, we aim to lower bound the number of queries that any quantum query algorithm makes to successfully output $\mathsf{F}(f) \in \Sigma$ for any input $f \in \mathsf{Func}$. All *decision* problems can be phrased in this form, where the set Σ is equal to $\{0,1\}$. However, it is not always possible to interpret more general *search* problems as computing a single-valued function $\mathsf{F}(f)$.

For instance, consider the simplest search problem, known as *Search*. If we restrict to the hardest inputs, all goes well: we have that each $f \in \mathsf{Func}$ is an n-bit string with Hamming weight 1, and $\mathsf{F}(f)$ is defined to be the unique index $i \in \Sigma = [n]$ such that $f(i) = 1$. However, if we relax Func to include all n-bit strings with Hamming weight at least 1, then there are multiple correct indices i such that $f(i) = 1$. Consequently, there is no longer a single correct value for $\mathsf{F}(f)$ for each $f \in \mathsf{Func}$. Further generalising Func to include all n-bit strings leads to cases where some inputs contain no indices mapping to 1, making $\mathsf{F}(f)$ undefined for such inputs.

In search problems, the problem is therefore characterised by a *relation* $\mathcal{R} \subset \mathsf{Func} \times \Sigma$, and the algorithm must output some $z \in \Sigma$ on input f such that $(f, z) \in \mathcal{R}$. This formulation generalises the concept of computing a function F, as we can define the relation $\mathcal{R}$ corresponding to F as the set $\{(f, \mathsf{F}(f)) : f \in \mathsf{Func}\}$. We shall see in Theorem 14 that the compressed oracle framework solves such search problems.

To remain closer to the notation used in Theorem 20, we still choose to frame search problems in terms of computing a function F. To accommodate the fact that search problems can have multiple, or even no, correct outputs, we define that a quantum query algorithm $\mathcal{A}$ has successfully computed a function F on an input $f \in \mathsf{Func}$ if it outputs z such that $z \in \mathsf{F}(f)$ (now allowed to be a *set* of valid outputs). Consequently, if Σ is the set of possible outputs, then each $\mathsf{F}(f)$ is a subset of Σ. To distinguish this from the earlier case where $\mathsf{F}(f)$ is a single value, we now write $\mathsf{F} : \mathsf{Func} \to 2^{\Sigma}$ for such search problems, where 2^{Σ} denotes the power set of Σ.

To reflect the modified definition of "success", we also update the projector F_z for each $z \in \Sigma$ in Theorem 10 to:

$$F_z = \sum_{\substack{f \in \mathsf{Func}: \\ \mathsf{F}(f) \ni z}} |f\rangle\langle f|.$$

We show that these modifications do not impact Item 3 in Theorem 10, thereby generalising Theorem 10 to search problems:

▶ **Lemma 12.** *Let Γ be a multiplicative adversary matrix for a problem $\mathsf{F} : \mathsf{Func} \to 2^{\Sigma}$ and let λ satisfy the constraints of Theorem 10. Let Λ_{bad} be the projector onto the eigenspaces of Γ corresponding to eigenvalues smaller than λ and let $\eta \leq 1 - \epsilon$ be a positive constant such that $\|F_z \Lambda_{\mathsf{bad}}\|^2 \leq \eta$ for every $z \in \Sigma$, where $F_z = \sum_{f \in \mathsf{Func}: \mathsf{F}(f) \ni z} |f\rangle\langle f|$.*

Then for any T-query quantum algorithm $\mathcal{A}$ that solves F on input $|\delta\rangle$ with success probability at least $1 - \epsilon$,

$$W^T(\Gamma, \mathcal{A}) \geq 1 + (\lambda - 1)\left(\sqrt{1-\epsilon} - \sqrt{\eta}\right)^2.$$

Proof. This and all subsequent proofs can be found in the full version [22]. ◀

3.3 The compressed oracle technique

In the compressed oracle technique [32], Zhandry observes that in query problems where the algorithm interacts with a quantum random oracle, it is equivalent (by applying a purification) to assume that the algorithm is run on a uniform superposition over all possible functions from the set X to the set Y. In this picture, a quantum adversary interacting with the quantum random oracle towards some nefarious end is analogous to a quantum algorithm run on input distribution δ, which is initialised to the uniform distribution over all functions from X to Y:

$$|\mathsf{Uniform}\rangle_{\mathcal{I}} \coloneqq \frac{1}{\sqrt{M^N}} \sum_{f \in Y^X} |f\rangle_{\mathcal{I}}. \tag{2}$$

We refrain from discussing the compressed oracle in depth here. For more details, see [32, 15, 20, 14]. Instead, we summarise the necessary parts needed to show how the compressed oracle technique can be used to derive quantum query lower bounds whenever $\mathsf{Func} = Y^X$. The input register $\mathcal{I}$ holding any computational basis state $|f\rangle_{\mathcal{I}}$, where $f \in Y^X$, can be viewed as a tensor product of the different function values for f for different values of $x \in X$:

$$|f\rangle_{\mathcal{I}} = \bigotimes_{x \in X} |f(x)\rangle_{\mathcal{I}_x}.$$

This can be interpreted as a look-up table that fully describes the action of f. We can also consider a Fourier basis (see Definition 5) for this register that represents a function in Y^X. Let $\{|\hat{f}\rangle\}_{f \in Y^X}$ be the *Fourier basis* of $\mathcal{I} \equiv \mathcal{Y}^{\otimes N}$, where each $|\hat{f}\rangle$ is defined as

$$|\hat{f}\rangle_{\mathcal{I}} \coloneqq \bigotimes_{x \in X} \mathsf{QFT}_M |f(x)\rangle_{\mathcal{I}_x} = \bigotimes_{x \in X} |\widehat{f(x)}\rangle_{\mathcal{I}_x}.$$

From this look-up table perspective, this means that we change the basis of all our entries in the look-up table. The key insight that Zhandry makes is that if we view both the input register $\mathcal{I}$ in this Fourier basis, as well as the $\mathcal{Y}$ register, then a query (as in Definition 4) acts on a basis state $|x\rangle_{\mathcal{X}}|\hat{y}\rangle_{\mathcal{Y}}|\hat{f}\rangle_{\mathcal{I}}$ as follows:

$$\mathcal{O}\left(|x\rangle_{\mathcal{X}}|\hat{y}\rangle_{\mathcal{Y}}|\hat{f}\rangle_{\mathcal{I}}\right) = |x\rangle_{\mathcal{X}}|\hat{y}\rangle_{\mathcal{Y}}|\widehat{f - y \cdot \delta_x}\rangle_{\mathcal{I}}. \tag{3}$$

Here, δ_x denotes the point function satisfying $\delta_x(x) = 1$ and $\delta_x(x') = 0$ for all $x' \neq x$, so $f - y \cdot \delta_x$ is the function that agrees with f on all values except possibly x, where it takes value $f(x) - y$. This change of perspective is quite peculiar: where in a regular query (as in Definition 4) the information stored in the $\mathcal{I}_x$ register is "copied" into the $\mathcal{Y}$ register, this interaction is mirrored when viewing the $\mathcal{I}_x$ register in the Fourier basis. Another added benefit of this basis change is that the initial state $|\mathsf{Uniform}\rangle$ simplifies to

$$|\mathsf{Uniform}\rangle_{\mathcal{I}} = \bigotimes_{x \in X} |\hat{0}\rangle_{\mathcal{I}_x}. \tag{4}$$

The action of the oracle in Equation (3), combined with Equation (4), implies the following consequence, which is the cornerstone of the compressed oracle technique:

▶ **Fact 13.** *For any T-query quantum algorithm $\mathcal{A}$ and for any $t \in [T]_0$, we have that $\rho^t_{\mathcal{I}}(\mathcal{A}, \mathsf{Uniform})$ is supported on vectors in the Fourier basis of the form $|\hat{f}\rangle$ where*

$$f = y_1 \cdot \delta_{x_1} + \cdots + y_s \cdot \delta_{x_s},$$

for some $x_1, \ldots, x_s \in X$, $y_1, \ldots y_s \in Y$, and $s \in [t]_0$.

In Lemma 18, we will establish a stronger relationship that directly implies Fact 13.

We can construct an isometry $\mathsf{Comp}_x : \mathbb{C}[Y] \to \mathbb{C}[Y \cup \{\bot\}]$, for every $x \in X$, that maps the $\mathcal{I}_x$ register to $|\bot\rangle$ if and only if this register contains $|\hat{0}\rangle$, which represents the algorithm knowing nothing about the value stored in register $\mathcal{I}_x$:

$$\mathsf{Comp}_x = |\bot\rangle\langle\hat{0}| + \sum_{z \in Y \setminus \{0\}} |\hat{z}\rangle\langle\hat{z}|.$$

By doing this for every $x \in X$ we obtain the isometry

$$\mathsf{Comp} = \bigotimes_{x \in X} \mathsf{Comp}_x.$$

This isometry Comp *compresses* the information of each of the basis vectors $|\hat{f}\rangle$, for $f = y_1\delta_{x_1} + \cdots + y_s\delta_{x_s}$, in the support of $\rho_{\mathcal{I}}^t(\mathcal{A}, \mathsf{Uniform})$, since $\mathsf{Comp}|\hat{f}\rangle \in \mathbb{C}[(Y \cup \{\bot\})^X]$ has $|\bot\rangle$ everywhere except for those $s \leq t$ registers indexed by $x_1, \ldots, x_s$. Let us extend QFT_M to $\mathbb{C}[(Y \cup \{\bot\})^X]$ by defining $\mathsf{QFT}_M|\bot\rangle = |\bot\rangle$. We can view

$$|D\rangle = \mathsf{QFT}_M^\dagger \mathsf{Comp}|\hat{f}\rangle \in \mathbb{C}[(Y \cup \{\bot\})^X]$$

as a *database*, where we have applied $\mathsf{QFT}_M^\dagger$ to bring the databases back to the computational basis. We say that D has size s if $|\{x \in X : D(x) \neq \bot\}| = s$, which we denote by $|D| = s$, and remark that by $\mathsf{QFT}_M|\bot\rangle = |\bot\rangle$, this basis conversion leaves the size of the database unaffected. We write

$$\mathcal{D}_s \coloneqq \{D \in (Y \cup \{\bot\})^X : |D| = s\}, \qquad \mathcal{D}_{\leq s} \coloneqq \{D \in (Y \cup \{\bot\})^X : |D| \leq s\}, \tag{5}$$

for the sets of all databases of size s and at most s, respectively. We let $\mathcal{D} = (Y \cup \{\bot\})^X$ denote the set of all databases of any size. In this work, we use set notation when working with databases:

- For any $x \in X, y \in Y$ and $D \in (Y \cup \{\bot\})^X$ such that $D(x) = \bot$, we can add a new entry (x, y) to D, to obtain $D' = D \cup (x, y)$. This means that the resulting database D' satisfies $D(x') = D'(x')$ for every $x' \in X \setminus \{x\}$ and $D'(x) = y$.
- For any $x \in X, y \in Y$ and $D \in (Y \cup \{\bot\})^X$ such that $D(x) = y$, we can delete the entry (x, y) from D, to obtain $D' = D \setminus (x, y)$. This means that the resulting database D' satisfies $D(x') = D'(x')$ for every $x' \in X \setminus \{x\}$ and $D'(x) = \bot$.

The compressed oracle gets its name from the fact that each database D of size s can be efficiently represented by the list of pairs $(x_1, D(x_1)), \ldots, (x_s, D(x_s))$, which is bounded in size due to Fact 13. Hence, the oracle operation $\mathcal{O}_{x,y}$ can be efficiently computed by a quantum algorithm that lazy samples from the uniform distribution, and this circuit (see [15] for its explicit construction) is referred to as the compressed (Fourier) oracle:

$$\mathsf{cO}_{x,y} = \mathsf{Comp} \circ \mathcal{O}_{x,y} \circ \mathsf{Comp}^\dagger. \tag{6}$$

This framework has many applications in cryptography [15, 26, 19, 16] by being able to analyse the interaction of an adversary with a random oracle, which as we have seen is equivalent to where the input register $\mathcal{I}$ is initialised to the uniform superposition over all functions (see Equation (2)). In [15, 20], it was shown that this can be generalised to superpositions over distributions where there is no correlation between the values in the registers $\mathcal{I}_x$ and $\mathcal{I}_{x'}$ for distinct $x, x' \in X$. In this work, we focus only on the application of the compressed oracle technique to quantum query lower bounds. A rigorous framework of this application has been given in [14], where the main ingredient of this lower bound (see Theorem 14 for the full statement) is of the following form:

$$\max_{x \in X, y \in Y} \left\| \mathsf{P}_{\mathcal{D}_\mathcal{P}} \mathsf{cO}_{x,y} \left(I - \mathsf{P}_{\mathcal{D}_\mathcal{P}}\right) \right\|.$$

Here, the property $\mathcal{P} \subseteq (X \times Y)^k$ defines a set of tuples of size k over $X \times Y$. Each tuple $p \in \mathcal{P}$ is an element of $(X \times Y)^k$ and represents a list of input-output pairs $((x_1, y_1), \ldots, (x_k, y_k))$. A property $\mathcal{P}$ induces a relation $\mathcal{R}$ on the input $f \in \mathsf{Func}$, as discussed in Section 3.2, by saying that for $p = (x_1, y_1), \ldots, (x_k, y_k) \in \mathcal{P}$, we have $(f, p) \in \mathcal{R}$ if and only if the input-output pairs in p are consistent with the input f. As an example, consider the collision problem, where for any input $f \in Y^X$, the goal is to output a pair (x_1, y), (x_2, y) such that $f(x_1) = f(x_2) = y$, referred to as a *collision*. The corresponding property $\mathcal{P}$ in this case would be

$$\mathcal{P} = \{((x_1, y_1), (x_2, y_2)) \in (X \times Y)^2 : y_1 = y_2\}.$$

A property $\mathcal{P}$ also induces a subset $\mathcal{D}_\mathcal{P} \subseteq \mathcal{D}$, where $D \in \mathcal{D}_\mathcal{P}$ if and only if it is consistent with one of the tuples in $\mathcal{P}$, meaning that there exists a $k \in [N]$ and $p = ((x_1, y_1), \ldots, (x_k, y_k)) \in \mathcal{P}$ such that $D(x_1) = y_1, \ldots, D(x_k) = y_k$. For any subset $A \subseteq \mathcal{D}$, we denote the projection onto this subset as

$$\mathsf{P}_A = \sum_{D \in A} |D\rangle\langle D|. \tag{7}$$

Since these projectors project onto computational basis states, we have the added benefit that they commute for distinct choices of A.

▶ **Theorem 14** ([14]). *Fix a finite set X of size N and let $Y = [M-1]_0$. Let $\mathcal{P} \subseteq (X \times Y)^k$ be a property for some $k \in [M-1]$ and consider a quantum algorithm $\mathcal{A}$ that outputs $(x_1, y_1), \ldots, (x_k, y_k)$. Let p be the probability that both $((x_1, y_1), \ldots, (x_k, y_k)) \in \mathcal{P}$ and $y_i = f(x_i)$ for every $i \in [k]$ when $\mathcal{A}$ has interacted with a random oracle, initialised with a uniformly random function f in Y^X. Then:*

$$\sqrt{p} \leq \sum_{t=1}^{T} \max_{x \in X, y \in Y} \left\| \mathsf{P}_{\mathcal{D}_{\leq t} \cap \mathcal{D}_\mathcal{P}} \mathsf{cO}_{x,y} \mathsf{P}_{\mathcal{D}_{\leq t-1} \setminus \mathcal{D}_\mathcal{P}} \right\| + \sqrt{\frac{k}{M}}.$$

▶ Remark 15. The framework in [14] allows for an adversary that makes both sequential as well as parallel queries, whereas we restrict to only the sequential query version of their result. Moreover, they also allow for a series of properties $\mathcal{P}_0, \ldots, \mathcal{P}_T$ instead of a single property $\mathcal{P}$, where they bound

$$\left\| \mathsf{P}_{\mathcal{D}_{\leq t} \cap \mathcal{D}_{\mathcal{P}_t}} \mathsf{cO}_{x,y} \mathsf{P}_{\mathcal{D}_{\leq t-1} \setminus \mathcal{D}_{\mathcal{P}_{t-1}}} \right\|.$$

Since the latter generalisation has thus far not been used for any application in the sequential query model, we consider the simplified lower bound as described and applied in [32, 25, 20].

The form of Theorem 14 is restricted compared to that of Theorem 10. We saw that we cannot run $\mathcal{A}$ on any input distribution, but only on $\mathsf{Uniform}$, since Theorem 14 requires the register $\mathcal{I}$ to be initialised with a uniformly random function f in Y^X. Since $\mathcal{A}$ has to output $((x_1, y_1), \ldots, (x_1, y_k)) \in \mathcal{P} \subseteq (X \times Y)^k$, the technique always deals with search problems instead of decision problems. Despite this restriction, it does seem to come with a large advantage compared to the adversary methods. In practice, it appears to be much more straightforward, or at least more intuitive, to come up with a good bound on $\|\mathsf{P}_{\mathcal{D}_{\leq t} \cap \mathcal{D}_\mathcal{P}} \mathsf{cO}_{x,y} \mathsf{P}_{\mathcal{D}_{\leq t-1} \setminus \mathcal{D}_\mathcal{P}}\|$ than it is to derive a good multiplicative adversary matrix Γ and accompanying constants λ, η, and bound its progress $\|\mathcal{O}_{x,y}^\dagger \Gamma^{1/2} \mathcal{O}_{x,y} \Gamma^{-1/2}\|$. Furthermore, like the multiplicative adversary method, it also works well when one considers exponentially small success probabilities, whereas the negative-weights adversary method fails in this regime.

3.4 Average-case query complexity

Theorem 14, as stated, does not explicitly give a lower bound on $Q_\epsilon(\mathsf{F})$, but it does imply one: Recall from Definition 7 and Section 3.2 that $Q_\epsilon(\mathsf{F})$ captures the number of queries required for any quantum query algorithm $\mathcal{A}$ to successfully output $z \in \mathsf{F}(f)$ for *any* input $f \in \mathsf{Func}$ with success probability at least $1-\epsilon$. By convexity, $Q_\epsilon(\mathsf{F})$ is lower bounded by the number of queries required for any input distribution δ, since:

$$\Pr_{f\sim\delta}[\mathcal{A} \text{ outputs } z \in \mathsf{F}(f)] \geq \min_{f\in\mathsf{Func}} \Pr[\mathcal{A} \text{ outputs } z \in \mathsf{F}(f)].$$

However, $Q_\epsilon(\mathsf{F})$ is not an interesting metric in the case where $\min_{f\in\mathsf{Func}} \Pr[\mathcal{A}$ outputs $z \in \mathsf{F}(f)]$ could be 0, i.e. when there exists an input $f \in \mathsf{Func}$ when the algorithm *can't* successfully output $z \in \mathsf{F}(f)$ for any input $f \in \mathsf{Func}$. This can occur in Theorem 14, as the input distribution δ is $\mathsf{Uniform}$. For instance, recall the collision problem. Some inputs $f \in Y^X$ may contain no collisions, making it impossible for the quantum algorithm to output $z \in \mathsf{F}(f)$.

Additionally, even if the worst-case input admits a non-zero probability of success, it can often be more meaningful to show that the problem is hard *on average* rather than merely demonstrating the existence of an input where the problem is hard. This is particularly relevant in the context of cryptography, where it is more desirable to know that a randomly chosen security key yields a secure construction than to prove that there exists a single specific key ensuring security. Therefore, in the remainder of this work, we focus on deriving a lower bound for the *average-case* complexity $Q^\delta_\epsilon(\mathsf{F})$ rather than the *worst-case* complexity $Q_\epsilon(\mathsf{F})$:

▶ **Definition 16** (ϵ-error Average-Case Quantum Query Complexity)**.** *Fix* $\mathsf{F} : \mathsf{Func} \to 2^\Sigma$*. Then the* ϵ-error average-case quantum query complexity *of* F *and input distribution* δ *on* Func*, denoted by* $Q^\delta_\epsilon(\mathsf{F})$*, is the minimum number of queries needed by any quantum query algorithm* $\mathcal{A}$ *such that*

$$\Pr_{f\sim\delta}[\mathcal{A} \textit{ outputs } z \in \mathsf{F}(f)] \geq 1-\epsilon.$$

We can now use Theorem 14 to lower bound $Q^{\mathsf{Uniform}}_\epsilon(\mathsf{F})$:

▶ **Corollary 17.** *Fix a finite set* X *of size* N *and let* $Y = [M-1]_0$*. Let* $\mathcal{P} \subseteq (X \times Y)^k$ *be a property for some* $k \in [M-1]$*. For every* $f \in Y^X$*, define*

$$\mathcal{P}(f) := \{((x_1, y_1), \ldots, (x_k, y_k)) \in \mathcal{P} : y_i = f(x_i) \textit{ for every } i \in [k]\}.$$

Then for any $\epsilon \in (0, 1-k/M)$ *and any problem* $\mathsf{F} : Y^X \to 2^{\mathcal{P}}$ *satisfying* $\mathsf{F}(f) \subseteq \mathcal{P}(f)$ *for every* $f \in Y^X$*, the* ϵ*-error average-case quantum query complexity* $Q^{\mathsf{Uniform}}_\epsilon(\mathsf{F})$ *is lower bounded by the smallest* T *satisfying*

$$\sqrt{1-\epsilon} - \sqrt{\frac{k}{M}} \leq \sum_{t=1}^{T} \max_{x\in X, y\in Y} \left\| \mathsf{P}_{\mathcal{D}_{\leq t}\cap\mathcal{D}_{\mathcal{P}}} \mathsf{cO}_{x,y} \mathsf{P}_{\mathcal{D}_{\leq t-1}\setminus\mathcal{D}_{\mathcal{P}}} \right\|.$$

Corollary 17 is slightly less conveniently phrased compared to Corollary 11 due to its dependence on t in the term

$$\left\| \mathsf{P}_{\mathcal{D}_{\leq t}\cap\mathcal{D}_{\mathcal{P}}} \mathsf{cO}_{x,y} \mathsf{P}_{\mathcal{D}_{\leq t-1}\setminus\mathcal{D}_{\mathcal{P}}} \right\|.$$

4 Multiplicative ladder adversary method

Here, we propose a simplified version of the multiplicative adversary method, that we name the *multiplicative ladder adversary* (MLA) method, which we later prove has the compressed oracle technique as a special case (see Section 5) as well as the polynomial method (see Section 7). The MLA method is weaker than the multiplicative adversary method as it only considers a subset of all possible multiplicative adversary matrices Γ, which we refer to as MLA matrices, but despite this restriction, it still exhibits a strong direct product theorem, as will be shown in Section 6.

4.1 Making the adversary matrix time-dependent

Before we define these MLA matrices in Definition 19, we first provide some motivation behind their definition. In Section 3.3, we saw that the compressed oracle seems to make more explicit use of the number of queries to compute the incremental progress by decomposing the set of all possible databases $\mathcal{D} = \bigsqcup_{t=0}^{N} \mathcal{D}_t$ based on their sizes and integrating these into the projection $\mathsf{P}_{\mathcal{D}_\mathcal{P}}$. We generalise this notion by introducing the following construction, that captures the subspace of $\mathbb{C}[Y^X]$ that is reachable from $|\delta\rangle$ after a fixed number of queries.

First, we define a few components necessary for our construction. Let δ be an initial distribution on $\mathsf{Func} \subseteq Y^X$. For any $t \in [N]$ and any choice of $x_1, \ldots, x_t \in X$ and $y_1, \ldots, y_t \in Y$, define

$$|v^{y_1,\ldots,y_t}_{x_1,\ldots,x_t}\rangle := \frac{1}{\sqrt{\alpha^{y_1,\ldots,y_t}_{x_1,\ldots,x_t}}} \sum_{\substack{f\in\mathsf{Func}:\\ \forall i\in[t], f(x_i)=y_i}} \sqrt{\delta(f)}|f\rangle, \tag{8}$$

where $\alpha^{y_1,\ldots,y_t}_{x_1,\ldots,x_t}$ is the normalisation factor, defined as

$$\alpha^{y_1,\ldots,y_t}_{x_1,\ldots,x_t} := \sum_{\substack{f\in\mathsf{Func}:\\ \forall i\in[t], f(x_i)=y_i}} \delta(f). \tag{9}$$

▶ **Lemma 18.** *Define the sequence of subspaces* $\mathsf{Space}_t(\delta)$ *as follows:*

- *For* $t = 0$*, let* $\mathsf{Space}_0(\delta) = \mathrm{span}\{|\delta\rangle\}$*, where* $|\delta\rangle = \sum_{f\in\mathsf{Func}} \sqrt{\delta(f)}|f\rangle$ *is the initial state of the input register* $\mathcal{I}$*.*
- *For* $t \in [N]$*, set*

$$\mathsf{Space}_t(\delta) := \mathrm{span}\left\{|v^{y_1,\ldots,y_t}_{x_1,\ldots,x_t}\rangle : (x_i, y_i) \in X \times Y \text{ for } i = 1, \ldots, t\right\}.$$

- *For* $t > N$*, define* $\mathsf{Space}_t(\delta) = \mathsf{Space}_N(\delta)$*.*

Then each space $\mathsf{Space}_t(\delta)$ *represents the subspace of* $\mathbb{C}[Y^X]$ *that is reachable from* $|\delta\rangle$ *after* t *queries:*

- *For every* $t \in [N]_0$*, there exists a* t*-query quantum algorithm* $\mathcal{A}$*, such that*

$$\mathsf{Space}_t(\delta) \subseteq \mathrm{supp}\left(\rho^t_\mathcal{I}(\mathcal{A}, \delta)\right).$$

- *For every* $t \in [N]_0$ *and* t*-query quantum algorithm* $\mathcal{A}$*, we have*

$$\mathsf{Space}_t(\delta) \supseteq \mathrm{supp}\left(\rho^t_\mathcal{I}(\mathcal{A}, \delta)\right).$$

Before we prove the lemma, we discuss some of its implications. First of all, we find that $\mathsf{Space}_N(\delta) = \mathrm{span}\{|f\rangle : f \in \mathrm{supp}(\delta)\}$. Moreover, in the special case where $|\delta\rangle = |\mathsf{Uniform}\rangle$, we have that

$$\mathsf{Space}_t(\mathsf{Uniform}) = \mathsf{Comp}^\dagger \left(\mathrm{span}\{|D\rangle : D \in \mathcal{D}_{\leq t}\}\right) \mathsf{Comp}, \tag{10}$$

which recovers Fact 13.

We can combine Γ with the projection $\Pi_{\leq t}$ that projects onto $\mathsf{Space}_t(\delta)$, to ensure that the progress keeps track of the number of queries done by the algorithm. The "$\leq$" in the subscript of each of the projectors $\Pi_{\leq t}$ is there to emphasise that $\Pi_{\leq t-1} \preceq \Pi_{\leq t}$. This is due to the fact that we can let $(x_t, y_t) = (x_{t-1}, y_{t-1})$ in $|v^{y_1,\ldots,y_t}_{x_1,\ldots,x_t}\rangle$. For any T-quantum algorithm $\mathcal{A}$, initial distribution δ, $t \in [T]_0$, and multiplicative adversary matrix Γ, we have

$$W^t(\Gamma, \mathcal{A}) = \mathrm{Tr}\left[\Gamma \rho^t_{\mathcal{I}}(\mathcal{A}, \delta)\right] = \mathrm{Tr}\left[\Gamma \Pi_{\leq t} \rho^t_{\mathcal{I}}(\mathcal{A}, \delta)\right] = \mathrm{Tr}\left[\Gamma \rho^t_{\mathcal{I}}(\mathcal{A}, \delta) \Pi_{\leq t}\right]. \tag{11}$$

4.2 Mapping the progress onto a ladder

The structure of the database projections $\mathsf{P}_{\mathcal{D}_{\leq t} \cap \mathcal{D}_{\mathcal{P}}}$ and $\mathsf{P}_{\mathcal{D}_{\leq t-1} \setminus \mathcal{D}_{\mathcal{P}}}$ in the compressed oracle technique (see Theorem 14) is, in practice, more convenient to work with than the more abstract projections Λ_i. This is because these projections are built from the database basis states, which are more intuitive and allow for easy tracking of their sizes with each query (see Fact 13).

We aim to establish a similar structure on the eigenspaces of Γ. These eigenspaces should resemble steps on a ladder, where each query moves the state up or down by at most one step. Additionally, these steps should be evenly spaced. To formalise this idea, we impose structural constraints on the spectral decomposition of Γ:

$$\Gamma = \sum_{i=0}^{\ell} \lambda_i \Lambda_i. \tag{12}$$

Here, $\ell + 1$ denotes the number of distinct eigenvalues of Γ, which are sorted in ascending order, and each Λ_i is the projector onto the eigenspace associated with the eigenvalue λ_i.

▶ **Definition 19** (Multiplicative Ladder Adversary Matrix). *Let $\Gamma = \sum_{i=0}^{\ell} \lambda_i \Lambda_i$ be a multiplicative adversary matrix. We say that Γ is a multiplicative ladder adversary (MLA) matrix if the following conditions hold:*

- *The eigenvalues of Γ satisfy $\lambda_i = \kappa^i$ for some $\kappa > 1$, so that*

$$\Gamma = \sum_{i=0}^{\ell} \kappa^i \Lambda_i.$$

- *For every $t \in [N]_0$, Γ commutes with $\Pi_{\leq t}$.*
- *For all $x \in \mathcal{X}$, $y \in \mathcal{Y}$, and $i, i' \in [\ell]_0$, the projections onto the eigenspaces satisfy*

$$\|\Lambda_{i'} \mathcal{O}_{x,y} \Lambda_i\| = 0, \quad \text{if } |i' - i| > 1. \tag{13}$$

The condition expressed in Equation (13) ensures that each query can move the state up or down by at most a single eigenspace. Meanwhile, the construction $\Gamma = \sum_{i=0}^{\ell} \kappa^i \Lambda_i$ ensures that the multiplicative progress between successive eigenspaces is constant, specifically a factor of κ.

This new definition allows us to prove an MLA-version of Theorem 10. This result is strictly weaker, as it only considers a subset of all possible multiplicative adversary matrices, but it greatly simplifies the upper bound on the progress achievable in a single query (Item 2).

▶ **Theorem 20.** *Fix a problem* $\mathsf{F} : \mathsf{Func} \to 2^{\Sigma}$*, an input distribution* δ *on* Func*, a constant* $\kappa > 1$*, and an MLA matrix* $\Gamma = \sum_{i=0}^{\ell} \kappa^i \Lambda_i$ *with* 1*-eigenstate* $|\delta\rangle$ *(see Definition 19). Let* λ *be a real number with* $1 < \lambda \leq \kappa^{\ell}$*. Let* Λ_{bad} *be the projector onto the eigenspaces of* Γ *corresponding to eigenvalues smaller than* λ *and let* $\eta \leq 1 - \epsilon$ *be a positive constant such that* $\|F_z \Lambda_{\mathsf{bad}}\|^2 \leq \eta$ *for every* $z \in \Sigma$*, where* $F_z = \sum_{\substack{f \in \mathsf{Func}: \\ \mathsf{F}(f) \ni z}} |f\rangle\langle f|$*. Then:*

1. *For any quantum algorithm* $\mathcal{A}$*,* $W^0(\Gamma, \mathcal{A}) = 1$*.*
2. *For any* T*-query quantum algorithm* $\mathcal{A}$*, and* $t \in [T-1]_0$*,*

$$\frac{W^{t+1}(\Gamma, \mathcal{A})}{W^t(\Gamma, \mathcal{A})} \leq \left(1 + \max_{\substack{i \in [\ell-1]_0, \\ x \in X, y \in Y}} \frac{\kappa - 1}{\sqrt{\kappa}} \|\Lambda_{i+1} \Pi_{\leq t+1} \mathcal{O}_{x,y} \Pi_{\leq t} \Lambda_i\|\right)^2$$

3. *For any* T*-query quantum algorithm* $\mathcal{A}$ *that solves* F *on input* $|\delta\rangle$ *with success probability at least* $1 - \epsilon$*,* $W^T(\Gamma, \mathcal{A}) \geq 1 + (\lambda - 1)(\sqrt{1-\epsilon} - \sqrt{\eta})^2$*.*

Note that the upper bound on W^{t+1}/W^t, the progress made in one step now depends on t. This is necessary to capture the power of the compressed oracle method, where, for example, the probability of (i.e. amplitude on) finding a collision in a single query is greater the more queried values you have stored in memory.

▶ **Corollary 21.** *For any* η *that satisfies the constraints of Theorem 20, any* $\epsilon \in (0, 1 - \eta)$*, problem* $\mathsf{F} : \mathsf{Func} \to 2^{\Sigma}$*, and input distribution* δ *on* Func*, the* ϵ*-error average-case quantum query complexity* $Q_{\epsilon}^{\delta}(\mathsf{F})$ *is lower bounded by the smallest* T *such that*

$$1 \leq \min_{\Gamma, \lambda} \Bigg(- (\lambda - 1)(\sqrt{1-\epsilon} - \sqrt{\eta})^2 + \prod_{t=1}^{T} \left(1 + \max_{\substack{i \in [\ell-1]_0, \\ x \in X, y \in Y}} \frac{\kappa - 1}{\sqrt{\kappa}} \|\Lambda_{i+1} \Pi_{\leq t} \mathcal{O}_{x,y} \Pi_{\leq t-1} \Lambda_i\|\right)^2\Bigg),$$

The machinery of MLA matrices is not necessary for the reduction in Section 5. For this reduction, we construct multiplicative matrices with $\ell = 1$, which automatically satisfy Equation (13). However, a general ℓ is required if we aim to compute a function F on ℓ independent instances simultaneously, as discussed in Section 6. Furthermore, almost all multiplicative adversary matrices constructed so far to establish lower bounds (see [7, 31, 6]) are, in fact, MLA matrices. This observation suggests that MLA matrices form a natural subset worthy of deeper analysis.

The following is a useful property, which we will employ in the subsequent sections:

▶ **Fact 22.** $\|\Lambda_i \Pi_{\leq t} \mathcal{O}_{x,y} \Pi_{\leq t-1} \Lambda_{i-1}\|$ *is monotonically non-decreasing in* $t \in [N]$ *for all* $i \in [\ell]$*.*

5 Reduction from the compressed oracle technique

In this section, we present an explicit reduction from the compressed oracle technique to our new MLA method:

▶ **Theorem 23.** *Fix a finite set* X *of size* N *and let* $Y = [M-1]_0$*. Consider a property* $\mathcal{P} \subseteq (X \times Y)^k$ *for some* $k \in [M-1]$*. For every* $f \in Y^X$*, define*

$$\mathcal{P}(f) := \{((x_1, y_1), \ldots, (x_k, y_k)) \in \mathcal{P} : y_i = f(x_i) \text{ for every } i \in [k]\}.$$

Let $\epsilon \in \left(0, 1-(9-4\sqrt{2})\frac{k}{M}\right)$*, and fix any problem* $\mathsf{F} : Y^X \to 2^{\mathcal{P}}$ *satisfying* $\mathsf{F}(f) \subseteq \mathcal{P}(f)$ *for every* $f \in Y^X$*. Define the quantities* $\mathsf{MLADV}^{\mathsf{Uniform}}_{\epsilon,\frac{2k}{M}}(\mathsf{F})$ *and* $\mathsf{COMP}^{\mathsf{Uniform}}_{\epsilon}(\mathsf{F})$ *as the lower bounds on* $Q^{\mathsf{Uniform}}_{\epsilon}(\mathsf{F})$ *obtained by Corollary 21 (with* η *set to* $\frac{2k}{M}$*) and Corollary 17, respectively. Then, we have*

$$\mathsf{COMP}^{\mathsf{Uniform}}_{\epsilon}(\mathsf{F}) \leq 6 \cdot \mathsf{MLADV}^{\mathsf{Uniform}}_{\epsilon,\frac{2k}{M}}(\mathsf{F}).$$

Recall from Corollary 17 that $\mathsf{COMP}^{\mathsf{Uniform}}_{\epsilon}(\mathsf{F})$ is equal to the smallest T satisfying

$$\sqrt{1-\epsilon} - \sqrt{\frac{k}{M}} \leq \sum_{t=1}^{T} \max_{x\in X, y\in Y} \left\| \mathsf{P}_{\mathcal{D}_{\leq t}\cap\mathcal{D}_{\mathcal{P}}} \mathsf{cO}_{x,y} \mathsf{P}_{\mathcal{D}_{\leq t-1}\setminus\mathcal{D}_{\mathcal{P}}} \right\|. \tag{14}$$

We start by removing the compressed oracle $\mathsf{cO}_{x,y}$ in Equation (14). For $t \in [T]_0$, consider the following projections:

$$\begin{aligned} \Pi_{1,t} &:= \mathsf{Comp}^{\dagger}\mathsf{P}_{\mathcal{D}_{\leq t}\cap\mathcal{D}_{\mathcal{P}}}\mathsf{Comp}, \\ \Pi_{0,t} &:= \mathsf{Comp}^{\dagger}\mathsf{P}_{\mathcal{D}_{\leq t}\setminus\mathcal{D}_{\mathcal{P}}}\mathsf{Comp}. \end{aligned} \tag{15}$$

By the definition of $\mathsf{cO}_{x,y}$ from Equation (6), we find that the projections in Equation (15) allow us to rewrite the right-hand side of Equation (14) as:

$$\begin{aligned} &\sum_{t=1}^{T} \max_{x\in X, y\in Y} \left\| \mathsf{P}_{\mathcal{D}_{\leq t}\cap\mathcal{D}_{\mathcal{P}}} \mathsf{cO}_{x,y} \mathsf{P}_{\mathcal{D}_{\leq t-1}\setminus\mathcal{D}_{\mathcal{P}}} \right\| \\ &= \sum_{t=1}^{T} \max_{x\in X, y\in Y} \left\| \left(\mathsf{Comp}\Pi_{1,t}\mathsf{Comp}^{\dagger}\right)\left(\mathsf{Comp}\mathcal{O}_{x,y}\mathsf{Comp}^{\dagger}\right)\left(\mathsf{Comp}\Pi_{0,t-1}\mathsf{Comp}^{\dagger}\right) \right\| \\ &= \sum_{t=1}^{T} \max_{x\in X, y\in Y} \left\| \Pi_{1,t}\mathcal{O}_{x,y}\Pi_{0,t-1} \right\|. \end{aligned}$$

Hence, by Corollary 17, $\mathsf{COMP}^{\mathsf{Uniform}}_{\epsilon}(\mathsf{F})$ is upper bounded by the smallest value of T satisfying

$$\sqrt{1-\epsilon} - \sqrt{\frac{k}{M}} \leq \sum_{t=1}^{T} \max_{x\in X, y\in Y} \left\| \Pi_{1,t}\mathcal{O}_{x,y}\Pi_{0,t-1} \right\|. \tag{16}$$

Next, we show that for any $\mathcal{P} \subseteq (X \times Y)^k$, we can always construct an explicit MLA Γ (see Definition 19), with accompanying parameter λ and $\eta = \frac{2k}{M}$ satisfying the conditions of Theorem 20, such that any T that satisfies

$$1 + (\lambda - 1)(\sqrt{1-\epsilon} - \sqrt{\eta})^2 \leq \prod_{t=1}^{T} \max_{\substack{i\in[\ell-1]_0 \\ x\in X, y\in Y}} \left(1 + \frac{\kappa-1}{\sqrt{\kappa}} \left\| \Lambda_{i+1}\Pi_{\leq t}\mathcal{O}_{x,y}\Pi_{\leq t-1}\Lambda_i \right\|\right)^2$$

also satisfies

$$\sqrt{1-\epsilon} - \sqrt{\frac{k}{M}} \leq \sum_{t=1}^{6T} \max_{x\in X, y\in Y} \left\| \Pi_{1,t}\mathcal{O}_{x,y}\Pi_{0,t-1} \right\|. \tag{17}$$

This then proves Theorem 23 by Corollary 21 and Equation (16).

For $\ell = 1$, we know from Definition 19 that any multiplicative ladder adversary matrix has the following form for some $\kappa > 1$:

$$\Gamma = \Lambda_0 + \kappa\Lambda_1.$$

We set the eigenspaces of Γ to correspond to the projections $\Lambda_1 \coloneqq \Pi_{1,N}$ (see Equation (15)) and $\Lambda_0 \coloneqq I - \Lambda_1$.

▷ Claim 24. For each $t \in [T]_0$

$$\Pi_{\leq t}\Lambda_0 = \Pi_{0,t}, \qquad \Lambda_1\Pi_{\leq t} = \Pi_{1,t}. \tag{18}$$

▷ Claim 25. Let $\Lambda_1 \coloneqq \Pi_{1,N}$ (see Equation (15)), $\Lambda_0 = I - \Lambda_1$ and $\Gamma = \Lambda_0 + \kappa\Lambda_1$ for some constant $\kappa > 1$. Then Γ is an MLA matrix as defined in Definition 19 with $|\mathsf{Uniform}\rangle$ as a 1-eigenvector.

Knowing that Γ is an MLA with $|\mathsf{Uniform}\rangle$ as a 1-eigenvector, we may apply Corollary 21. By taking the natural logarithm of both sides, it states

$$\ln\left(1 + (\lambda - 1)(\sqrt{1-\epsilon} - \sqrt{\eta})^2\right) \leq 2\sum_{t=1}^{T} \ln\left(1 + \max_{x\in X, y\in Y} \frac{\kappa - 1}{\sqrt{\kappa}} \left\|\Lambda_1\Pi_{\leq t}\mathcal{O}_{x,y}\Pi_{\leq t-1}\Lambda_0\right\|\right).$$

To show that this implies Equation (16), we set $\lambda = \kappa = 1 + (e-1)/\left(\sqrt{1-\epsilon} - \sqrt{\eta}\right)^2$ and multiply both sides of the equation with $\sqrt{1-\epsilon} - \sqrt{\eta}$ to arrive at

$$\begin{aligned}
\sqrt{1-\epsilon} - \sqrt{\eta} &\leq 2\left(\sqrt{1-\epsilon} - \sqrt{\eta}\right)\sum_{t=1}^{T} \ln\left(1 + \max_{x\in X, y\in Y} \frac{\kappa - 1}{\sqrt{\kappa}} \left\|\Lambda_1\Pi_{\leq t}\mathcal{O}_{x,y}\Pi_{\leq t-1}\Lambda_0\right\|\right) \\
&\leq 2\left(\sqrt{1-\epsilon} - \sqrt{\eta}\right)\frac{\kappa - 1}{\sqrt{\kappa}}\sum_{t=1}^{T} \max_{x\in X, y\in Y} \left\|\Lambda_1\Pi_{\leq t}\mathcal{O}_{x,y}\Pi_{\leq t-1}\Lambda_0\right\| \\
&\leq 3\sum_{t=1}^{T} \max_{x\in X, y\in Y} \left\|\Lambda_1\Pi_{\leq t}\mathcal{O}_{x,y}\Pi_{\leq t-1}\Lambda_0\right\|. \qquad (19)
\end{aligned}$$

To finalise the proof, we show that the choice of $\eta = \frac{2k}{M}$ satisfies the conditions of Theorem 20. By our choice of Γ, λ, κ, the projection Λ_{bad} is equal to Λ_0. The proof of the following lemma can be skipped if the reader is familiar with the compressed oracle technique, as the technique is reminiscent to the proof of the lemma in [32] that links the compressed Fourier oracle to the original oracle.

▶ **Lemma 26.** *Let $\Gamma = \Lambda_0 + \kappa\Lambda_1$ be a multiplicative adversary matrix (see Definition 8) with $\Lambda_1 = \mathsf{Comp}^\dagger\mathsf{P}_{\mathcal{D}_\mathcal{P}}\mathsf{Comp}$ and $\Lambda_0 = I - \Lambda_1$. Then for every $z \in \mathcal{P} \subseteq (X \times Y)^k$ we have*

$$\|F_z\Lambda_0\|^2 \leq \frac{2k}{M},$$

where $F_z = \sum_{f\in Y^X : \mathsf{F}(f)\ni z} |f\rangle\langle f|$.

Knowing that $\frac{2k}{M}$ is a valid value for η, suppose that $\epsilon \leq 1 - (9 - 4\sqrt{2})\frac{k}{M}$. Then

$$\sqrt{1-\epsilon} - \left(2\sqrt{2} - 1\right)\sqrt{\frac{k}{M}} \geq 0.$$

Together with Claim 24, Equation (19) and Lemma 26, this means that our MLA matrix Γ satisfies Equation (17), where in the penultimate step we use that $\|\Lambda_1\Pi_{\leq t}\mathcal{O}_{x,y}\Pi_{\leq t-1}\Lambda_0\|$ is monotonically non-decreasing in t (see Fact 22):

$$\begin{aligned}
&\sqrt{1-\epsilon} - \sqrt{\frac{k}{M}} \leq \sqrt{1-\epsilon} - \sqrt{\frac{k}{M}} + \sqrt{1-\epsilon} - (2\sqrt{2}-1)\sqrt{\frac{k}{M}} = 2\left(\sqrt{1-\epsilon} - \sqrt{\eta}\right) \\
&\leq \sum_{t=1}^{6T} \max_{x\in X, y\in Y} \left\|\Lambda_1\Pi_{\leq t}\mathcal{O}_{x,y}\Pi_{\leq t-1}\Lambda_0\right\| \leq \sum_{t=1}^{6T} \max_{x\in X, y\in Y} \left\|\Pi_{1,t}\mathcal{O}_{x,y}\Pi_{0,t-1}\right\|.
\end{aligned}$$

6 A strong direct product theorem

The machinery of MLA matrices seems a bit overcomplicated compared to what we actually needed in the reduction in Section 5. Since we only considered multiplicative adversary matrices where $\ell = 1$, we obtain the "ladder" property automatically. We will need general ℓ however if we want to compute a function F on ℓ independent instances simultaneously.

Although it does not seem to fit in the framework of [14] directly, the compressed oracle framework also has the powerful property of being able to exhibit *strong direct product theorems* (SDPT), as shown in [25, 20]. Such a theorem states that if we try to compute F on k independent inputs in fewer queries than k times the queries needed for a single instance of F, then our success probability will decrease exponentially in k.

It was already shown by [31] that the multiplicative adversary method directly satisfies a SDPT. Here we show that a similar proof as in [6], which is based on the proof in [31], also holds for the MLA method due to the fact (which we will prove) that the set of MLA matrices is closed under tensor powers. This motivates the study of the MLA method as a simplification of the multiplicative adversary method, since it maintains the property of satisfying a SDPT.

We introduce the following notation for this section: for any problem $\mathsf{F} : \mathsf{Func} \to 2^{\Sigma}$ and integer $k \geq 1$ let $\mathsf{F}^{(k)} : \mathsf{Func}^k \to (2^{\Sigma})^k$ be defined as

$$\mathsf{F}^{(k)}(h_1, \ldots, h_k) = (\mathsf{F}(h_1), \ldots, \mathsf{F}(h_k)).$$

▶ **Theorem 27.** *For any problem $\mathsf{F} : \mathsf{Func} \to 2^{\Sigma}$, input distribution δ on Func, and fixed $\eta \leq \frac{1}{2}$, let $\mathsf{MLADV}^{\delta}_{\epsilon,\eta}(\mathsf{F})$ be the lower bound on $Q^{\delta}_{\epsilon}(\mathsf{F})$ obtained by Corollary 21. Then there exists a constant $c \in (0,1)$ such that for any integer $k > 361$ we have*

$$\mathsf{MLADV}^{\delta^k}_{1-c^k,\eta^{\frac{2k}{5}}}(\mathsf{F}^{(k)}) \geq \frac{k}{10}\mathsf{MLADV}^{\delta}_{1-\epsilon,\eta}(\mathsf{F}).$$

7 Reduction from the polynomial method

In this section, we show how we can reduce the polynomial method to our new MLA method. Note that in this section, we can revert to the original notion of "success" (see Definition 7 and Definition 16). The polynomial method, due to [9], allows for lower bounding the quantum query complexity of a boolean function F via its approximate degree:

▶ **Definition 28** (Approximate degree)**.** *For any $\epsilon \geq 0$, the approximate degree $\widetilde{\deg}_{\epsilon}(\mathsf{F})$ of a boolean function $\mathsf{F} : \{0,1\}^n \mapsto \{0,1\}$ is defined as*

$$\widetilde{\deg}_{\epsilon}(\mathsf{F}) = \min_{p} \{\deg(\mathsf{F}) : \forall x \in \{0,1\}^n, |p(x) - \mathsf{F}(x)| \leq \epsilon\}, \tag{20}$$

where the minimum is taken over all n-variate polynomials $p : \mathbb{R}^n \mapsto \mathbb{R}$.

▶ **Theorem 29** ([9])**.** *For any Boolean function F, we have $Q_{\epsilon}(\mathsf{F}) \geq \Omega(\widetilde{\deg}_{\epsilon}(\mathsf{F}))$.*

7.1 A tighter output condition

Recall from Theorem 10 that our progress measure in the multiplicative adversary framework must satisfy the following condition, from [31, 6]:

$$W^T(\Gamma, \mathcal{A}) \geq 1 + (\lambda - 1)\left(\sqrt{1-\epsilon} - \sqrt{\eta}\right)^2 \tag{21}$$

whenever $\mathcal{A}$ has error at most ϵ. To prove the reduction, we need to use the fact that the progress measure must satisfy an even stronger condition, due to [24, 27], which we now describe.

▶ **Definition 30** ((Hadamard product) fidelity)**.** *The fidelity $\mathcal{F}(\rho,\sigma)$ between two density matrices ρ and σ is defined as*

$$\mathcal{F}(\rho,\sigma) := \mathrm{Tr}\left[\sqrt{\sqrt{\rho}\sigma\sqrt{\rho}}\right].$$

The Hadamard product fidelity $\mathcal{F}(\rho,\sigma)$ (introduced in [27]) between two Gram matrices A and B is defined as

$$\mathcal{F}_H(A,B) := \min_{|u\rangle:\||u\rangle\|=1} \mathcal{F}\left(A\circ|u\rangle\langle u|, B\circ|u\rangle\langle u|\right),$$

where $\circ$ denotes the Hadamard (entrywise) product.

Let M be the Gram matrix corresponding to our function F, i.e.

$$M = \sum_{z\in\Sigma} \sum_{f,f'\in\mathsf{Func}:\mathsf{F}(f)=\mathsf{F}(f')=z} |f\rangle\langle f'|.$$

Then in [24, 27] it is shown that the condition

$$W^T(\Gamma,\mathcal{A}) \geq \min_N \left\{\mathrm{Tr}[\Gamma N] : \mathcal{F}_H(N,M) \geq \sqrt{1-\epsilon}, N\succeq 0, N\circ I = I\right\} \tag{22}$$

must be satisfied when Γ is as in Theorem 10, for any quantum algorithm $\mathcal{A}$ that solves F on input $|\delta\rangle$ with success probability at least $1-\epsilon$. This output condition is stronger than the one from Equation (21):

▶ **Fact 31.** *Let Γ be a multiplicative adversary matrix for a problem $\mathsf{F}:\mathsf{Func}\to 2^\Sigma$ with Gram matrix M and let λ satisfy the constraints of Theorem 10. Let Λ_{bad} be the projector onto the eigenspaces of Γ corresponding to eigenvalues smaller than λ and let $\eta\leq 1-\epsilon$ be a positive constant such that $\|F_z\Lambda_{\mathsf{bad}}\|^2\leq\eta$ for every $z\in\Sigma$, where $F_z=\sum_{f\in\mathsf{Func}:z=\mathsf{F}(f)}|f\rangle\langle f|$. Then for every gram matrix N s.t. $\mathcal{F}_H(N,M)\geq\sqrt{1-\epsilon}$, we have*

$$\mathrm{Tr}[\Gamma N] \geq 1+(\lambda-1)\left(\sqrt{1-\epsilon}-\sqrt{\eta}\right)^2.$$

This stronger output condition was used in [24] to exhibit an SDPT for quantum query complexity and in [27] for the reduction from the polynomial method to the multiplicative adversary method. However, due to its abstract phrasing, it is less suited to prove explicit lower bounds. It is straightforward to reprove our SDPT from Theorem 27 for this stronger output condition, following the same argument as in [27].

Under the output condition from Equation (22), we obtain the following strengthening of Corollary 21:

▶ **Corollary 32.** *For any $\epsilon\in(0,1]$, problem $\mathsf{F}:\mathsf{Func}\to\Sigma$ with Gram matrix M, the ϵ-error quantum query complexity $Q_\epsilon(\mathsf{F})$ is lower bounded by the smallest T such that*

$$\min_{\substack{N:\mathcal{F}_H(N,M)\geq\sqrt{1-\epsilon},\\ N\succeq 0, N\circ I=I}} \mathrm{Tr}[\Gamma N] \leq \min_\Gamma \prod_{t=1}^{T}\left(1+\max_{\substack{i\in[\ell-1]_0,\\ x\in X,y\in Y}} \frac{\kappa-1}{\sqrt{\kappa}}\|\Lambda_{i+1}\Pi_{\leq t}\mathcal{O}_{x,y}\Pi_{\leq t-1}\Lambda_i\|\right)^2.$$

The reduction then takes the following form, which is proven in the full version [22]:

▶ **Theorem 33.** *Fix any $\epsilon\in(0,1]$ and problem $\mathsf{F}:\{0,1\}^n\to\{0,1\}$. Let $\mathsf{MLADV}_\epsilon(\mathsf{F})$ be the lower bound on $Q_\epsilon(\mathsf{F})$ obtained by Corollary 32. Then, we have*

$$\widetilde{\deg}_\epsilon(\mathsf{F}) \leq 4\cdot\mathsf{MLADV}_\epsilon(\mathsf{F}).$$

References

1 Scott Aaronson and Yaoyun Shi. Quantum lower bounds for the collision and the element distinctness problems. *Journal of the ACM*, 51(4):595–605, 2004. arXiv: quant-ph/0112086 doi:10.1145/1008731.1008735.

2 Gorjan Alagic, Joseph Carolan, Christian Majenz, and Saliha Tokat. The sponge is quantum indifferentiable. *arXiv preprint*, 2025. arXiv:2504.16887.

3 A. Ambainis. Quantum lower bounds by quantum arguments. *Journal of Computer and System Sciences*, 64(4):750–767, 2002. Earlier version in STOC'00. arXiv: quant-ph/0002066 doi:10.1006/jcss.2002.1826.

4 Andris Ambainis. Polynomial degree vs. quantum query complexity. *Journal of Computer and System Sciences*, 72(2):220–238, 2006. arXiv: quant-ph/0305028 doi:10.1016/J.JCSS.2005.06.006.

5 Andris Ambainis. A new quantum lower bound method, with an application to a strong direct product theorem for quantum search. *Theory of Computing*, 6(1):1–25, 2010. arXiv: quant-ph/0508200 doi:10.4086/TOC.2010.V006A001.

6 Andris Ambainis, Loïck Magnin, Martin Roetteler, and Jérémie Roland. Symmetry-assisted adversaries for quantum state generation. In *2011 IEEE 26th Annual Conference on Computational Complexity*, pages 167–177. IEEE, 2011. arXiv: 1012.2112 doi:10.1109/CCC.2011.24.

7 Andris Ambainis, Robert Špalek, and Ronald de Wolf. A new quantum lower bound method, with applications to direct product theorems and time-space tradeoffs. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of Computing*, pages 618–633, 2006. arXiv: quant-ph/0511200

8 Howard Barnum and Michael Saks. A lower bound on the quantum query complexity of read-once functions. *Journal of Computer and System Sciences*, 69(2):244–258, 2004. arXiv: quant-ph/0201007 doi:10.1016/J.JCSS.2004.02.002.

9 Robert Beals, Harry Buhrman, Richard Cleve, Michele Mosca, and Ronald de Wolf. Quantum lower bounds by polynomials. *Journal of the ACM*, 48(4):778–797, 2001. Earlier version in FOCS'98. arXiv: quant-ph/9802049 doi:10.1145/502090.502097.

10 Aleksandrs Belovs. A direct reduction from the polynomial to the adversary method. In *19th Conference on the Theory of Quantum Computation, Communication and Cryptography*, 2024. arXiv: 2301.10317

11 Aleksandrs Belovs and Ansis Rosmanis. Adversary lower bounds for the collision and the set equality problems. *Quantum Information and Computation*, 2017. arXiv: 1310.5185

12 Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Sponge functions. In *ECRYPT hash workshop*, 2007.

13 Harry Buhrman and Robert Špalek. Quantum verification of matrix products. In *Proceedings of the 17th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 880–889, 2006. arXiv: quant-ph/0409035

14 Kai-Min Chung, Serge Fehr, Yu-Hsuan Huang, and Tai-Ning Liao. On the compressed-oracle technique, and post-quantum security of proofs of sequential work. In *Advances in Cryptology–EUROCRYPT 2021: 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17–21, 2021, Proceedings, Part II*, pages 598–629. Springer, 2021. ePrint: 2020/1305 doi:10.1007/978-3-030-77886-6_21.

15 Jan Czajkowski, Christian Majenz, Christian Schaffner, and Sebastian Zur. Quantum lazy sampling and game-playing proofs for quantum indifferentiability. *arXiv preprint*, 2019. arXiv:1904.11477.

16 Jelle Don, Serge Fehr, Christian Majenz, and Christian Schaffner. Online-extractability in the quantum random-oracle model. In *Advances in Cryptology–EUROCRYPT 2022: 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Trondheim, Norway, May 30–June 3, 2022, Proceedings, Part III*, pages 677–706. Springer, 2022. ePrint: 2021/280 doi:10.1007/978-3-031-07082-2_24.

17 Sebastian Dörn and Thomas Thierauf. The quantum query complexity of algebraic properties. In *Fundamentals of Computation Theory: 16th International Symposium, FCT 2007, Budapest, Hungary, August 27-30, 2007. Proceedings 16*, pages 250–260. Springer, 2007. arXiv: 0705.1446 doi:10.1007/978-3-540-74240-1_22.

18 Christoph Dürr, Mark Heiligman, Peter Høyer, and Mehdi Mhalla. Quantum query complexity of some graph problems. *SIAM Journal on Computing*, 35(6):1310–1328, 2006. Earlier version in ICALP'04. arXiv: quant-ph/0401091 doi:10.1137/050644719.

19 Alex B Grilo, Kathrin Hövelmanns, Andreas Hülsing, and Christian Majenz. Tight adaptive reprogramming in the qrom. In *Advances in Cryptology–ASIACRYPT 2021: 27th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 6–10, 2021, Proceedings, Part I 27*, pages 637–667. Springer, 2021. arXiv: 2010.15103 doi:10.1007/978-3-030-92062-3_22.

20 Yassine Hamoudi and Frédéric Magniez. Quantum time–space tradeoff for finding multiple collision pairs. *ACM Transactions on Computation Theory*, 15(1-2):1–22, 2023. arXiv: 2002.08944 doi:10.1145/3589986.

21 Peter Høyer, Troy Lee, and Robert Špalek. Negative weights make adversaries stronger. In *Proceedings of the 39th ACM Symposium on the Theory of Computing (STOC)*, pages 526–535, 2007. arXiv: quant-ph/0611054 doi:10.1145/1250790.1250867.

22 Stacey Jeffery and Sebastian Zur. The compressed oracle is a worthy (multiplicative) adversary. *arXiv preprint*, 2025. arXiv:2509.07876.

23 Hartmut Klauck, Robert Špalek, and Ronald De Wolf. Quantum and classical strong direct product theorems and optimal time-space tradeoffs. *SIAM Journal on Computing*, 36(5):1472–1493, 2007. arXiv: quant-ph/0402123 doi:10.1137/05063235X.

24 Troy Lee and Jérémie Roland. A strong direct product theorem for quantum query complexity. *computational complexity*, 22:429–462, 2013. arXiv: 1104.4468 doi:10.1007/S00037-013-0066-8.

25 Qipeng Liu and Mark Zhandry. On finding quantum multi-collisions. In *Advances in Cryptology–EUROCRYPT 2019: 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19–23, 2019, Proceedings, Part III 38*, pages 189–218. Springer, 2019. ePrint: 2018/1096 doi:10.1007/978-3-030-17659-4_7.

26 Qipeng Liu and Mark Zhandry. Revisiting post-quantum fiat-shamir. In *Advances in Cryptology – CRYPTO 2019: 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18–22, 2019, Proceedings, Part II 39*, pages 326–355. Springer, 2019. ePrint: 2019/262 doi:10.1007/978-3-030-26951-7_12.

27 Loïck Magnin and Jérémie Roland. Explicit relation between all lower bound techniques for quantum query complexity. *International Journal of Quantum Information*, 13(04):1350059, 2015. arXiv: 1209.2713

28 Ben W Reichardt. Span programs and quantum query complexity: The general adversary bound is nearly tight for every boolean function. In *2009 50th Annual IEEE Symposium on Foundations of Computer Science*, pages 544–551. IEEE, 2009. arXiv: 0904.2759 doi:10.1109/FOCS.2009.55.

29 Ansis Rosmanis. Tight bounds for inverting permutations via compressed oracle arguments. *arXiv preprint*, 2021. arXiv:2103.08975.

30 Alexander A Sherstov. Strong direct product theorems for quantum communication and query complexity. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 41–50, 2011. arXiv: 1011.4935 doi:10.1145/1993636.1993643.

31 Robert Špalek. The multiplicative quantum adversary. In *2008 23rd Annual IEEE Conference on Computational Complexity*, pages 237–248. IEEE, 2008. arXiv: quant-ph/0703237

32 Mark Zhandry. How to record quantum queries, and applications to quantum indifferentiability. In *Annual International Cryptology Conference*, pages 239–268. Springer, 2019. ePrint: 2018/276 doi:10.1007/978-3-030-26951-7_9.

33 Shengyu Zhang. On the power of ambainis lower bounds. *Theoretical Computer Science*, 339(2-3):241–256, 2005. arXiv: quant-ph/0311060 doi:10.1016/J.TCS.2005.01.019.