A quantum while loop for amplitude amplification

Pablo Andrés-Martínez
Chris Heunen

September 18, 2020

Abstract

Grover's algorithm searches an unstructured database of N entries, finding a marked element in time $\mathcal{O}\left(\sqrt{N}\right)$. The algorithm was later generalised to amplitude amplification. The original algorithm succeeds with probability close to one and needs to run for a number of iterations chosen ahead of time. We discuss an implementation of amplitude amplification that, instead, runs for a number of iterations unspecified a priori and only halts on success. We prove that this approach maintains the quadratic speed-up of the original quantum algorithm. The key component is a feedback loop controlled by a weak measurement that allows limited monitoring of the evolution of the quantum state.

In classical computer science, it is essential to have loops whose number of iterations is not determined a priori, but instead terminate when (and if) the internal state reaches some chosen condition. There are various ways to construct such an iterating structure, general recursion and while loops being the most common ones. A programming language that allows such a construction as well as natural numbers is automatically Turing-complete [7], meaning that any computable function can be expressed in the language. In contrast, loops whose number of iterations is determined beforehand are provably less powerful: these loops together with the natural numbers do not suffice for Turing-completeness.

In quantum computer science, *while loops* usually do not hold centre stage. They are used often, but nearly always in a trivial way, for instance, when we say that a quantum algorithm is repeated until it succeeds. We will refer to this mundane use of while loops as *test-restart loops.* Think of a classical computer controlling a quantum chip, checking whether the outcome of the quantum algorithm was successful and, if not, instructing the quantum chip to run it again – there is nothing quantum about the while loop itself.

This article implements the amplitude amplification quantum algorithm with a while loop at its core. Whereas the standard approach runs the algorithm for a predetermined number of iterations and then performs a projective measurement, our approach applies a *weak measurement* on the quantum state in every iteration and stops only when success is certain. Such a while loop is arguably more interesting because we are, in some sense, monitoring an evolving quantum state in real time. Perhaps surprisingly, the quantum speed-up is maintained: on average, our approach requires a number of iterations quadratically smaller than any known classical algorithm. This is possible because the 'monitoring' is achieved via weak measurements, which do not fully collapse the quantum state, but also provide little information on each application.

This approach was first proposed by Mizel [15] in 2008 with a treatment that is different from ours. In particular, we make the connection to while loops [20] and to weak measurements [5]. Furthermore, we take a different route to prove (in Section 2) that this approach yields the same time-complexity as the standard amplitude amplification algorithm. Understanding our proof requires no knowledge of mixed quantum states; instead, elementary trigonometry and a brief introduction to weak measurements (in Section 1.3) suffice. This simpler presentation is accessible to a larger audience and gives more insight on the evolution of the quantum state throughout the algorithm.

We also show (in Section 3) that our approach (as well as Mizel's) has essentially the same behaviour as a version of the algorithm that uses a simple test-restart loop. This means that our notion of while loop provides no algorithmic advantage. Instead, our results should be regarded as a step towards a better understanding of the role of while loops in quantum computer science. In the long term, this programme aims to facilitate development of quantum algorithms by identifying useful primitives and programming constructs that are native to quantum algorithms rather than classical ones.

1 Background

The main result presented in this paper (Section 2) was first proposed by Mizel [15], although we arrived at it independently, and explore it from a different perspective and provide simpler proofs. Mizel considers evolving a quantum state through standard amplitude amplification, measuring it on every iteration to check if the algorithm has succeeded. However, due to technical reasons we discuss in Section 1.3, directly applying projective measurements would prevent the state from evolving as intended, causing the loss of the quantum speed-up. Instead, Mizel indirectly monitors the state by including an auxiliary qubit that, on every iteration, is measured after interacting weakly with the system we wish to observe. In

Section 1.3, we describe this procedure as the typical example of a weak measurement.

1.1 Amplitude amplification

Amplitude amplification is a generalisation of Grover's search algorithm [11]. In Grover's problem, we are given an unsorted set of elements B, about which we know no structure or heuristics, and a function $\chi\colon B \to \{0,1\}$ that satisfies $\chi(\star) = 1$ for only one element $\star \in B$. We are tasked with finding this marked element $\star$. Grover's algorithm does so in $\mathcal{O}\left(\sqrt{|B|}\right)$ calls to χ. In contrast, the any classical algorithm requires $\frac{|B|}{2}$ calls to χ on average: knowing no information about B, your only choice is to try each element one by one. Considering that searching a dataset is a fundamental primitive on a broad range of fields, this algorithm is one of the most promising applications of quantum computers.

Amplitude amplification [4] generalises Grover's problem in two ways: there may be multiple marked elements and it allows us to make use of a procedure $\mathcal{A}$ that may find a marked element with higher chance than picking elements of B at random, *i.e.* it allows us to exploit knowledge about B. The algorithm takes this procedure $\mathcal{A}$ and amplifies the probability of it yielding a marked element, so that it is found quadratically faster. A formal definition is provided below.

Definition 1.1. *Let B be a finite set spanning a Hilbert space $\mathcal{H} = \mathrm{span}(B)$ and let $\chi\colon B \to \{0,1\}$ be a function that characterises the marked subset of B. Define Hilbert spaces*

$$\mathcal{H}_i = \mathrm{span}\{b \in B \mid \chi(b) = i\}$$

for $i \in \{0,1\}$ and $\mathcal{H} = \mathcal{H}_0 \oplus \mathcal{H}_1$. Furthermore, choose some $|\psi\rangle \in \mathcal{H}$ as the initial state *and assume there exists a procedure described by a unitary $\mathcal{A}$ such that $|\psi\rangle = \mathcal{A}|0\rangle$. An algorithm performs* amplitude amplification *if, starting from the initial state $|\psi\rangle$, it returns a state in $\mathcal{H}_1$ with probability close to 1.*

Write P_1 for the orthogonal projection onto $\mathcal{H}_1$ and

$$\rho = \langle\psi|P_1|\psi\rangle \tag{1}$$

for the initial success probability. The algorithm is efficient *if its expected number of calls to an oracle implementing χ is $\mathcal{O}\left(1/\sqrt{\rho}\right)$.*

Grover's problem is recovered from Definition 1.1 by making $\dim(\mathcal{H}_1) = 1$ and choosing $\mathcal{A}$ to generate a state $|\psi\rangle$ that is the uniform superposition of all elements of the basis B.

1.2 Standard amplitude amplification algorithm

This section describes the standard algorithm [4] for amplitude amplification. We use the notation introduced in Definition 1.1. Let $|\psi_0\rangle \in \mathcal{H}_0$ and $|\psi_1\rangle \in \mathcal{H}_1$ be the normalised projections of $|\psi\rangle$ onto $\mathcal{H}_0$ and $\mathcal{H}_1$. Then,

$$|\psi\rangle = \cos\alpha|\psi_0\rangle + \sin\alpha|\psi_1\rangle \tag{2}$$

for some an angle $\alpha \in [0, \frac{\pi}{2}]$. As in Definition 1.1, the initial success probability is

$$\rho = \langle\psi|P_1|\psi\rangle = \sin^2\alpha. \tag{3}$$

Amplitude amplification is meant to be used when the initial probability of success ρ is small, then $\alpha = \arcsin\sqrt{\rho}$ may be approximated as $\alpha \approx \sqrt{\rho}$.

Given any state $|\phi\rangle \in \mathcal{H}$ we refer to its reflection operator as S_ϕ, given by

$$S_\phi = 2|\phi\rangle\langle\phi| - I_{\mathcal{H}}. \tag{4}$$

Define an iteration of amplitude amplification as the operator Q_χ

$$Q_\chi = S_\psi\, O_\chi \tag{5}$$

$$O_\chi = \sum_{b\in B}(-1)^{\chi(b)}\,|b\rangle\langle b| \tag{6}$$

where O_χ is interpreted as a single call to the oracle function χ. S_ψ may be implemented using procedure $\mathcal{A}$:

$$S_\psi = \mathcal{A}S_0\mathcal{A}^{-1} \tag{7}$$

where $\mathcal{A}$ is a unitary operator by Definition 1.1.

Consider any state $|\phi\rangle$ of the form

$$|\phi\rangle = \cos a|\psi_0\rangle + \sin a|\psi_1\rangle \tag{8}$$

for some $a \in [0, 2\pi)$. On these states, the action of Q_χ simply corresponds to a reflection over ψ followed by a reflection over ψ_0. Thus, a simple geometry exercise shows that the state after applying Q_χ will remain in the plane spanned by $|\psi_0\rangle$ and $|\psi_1\rangle$, increasing the angle a above by 2α. In general, for any number $k \in \mathbb{N}$,

$$Q_\chi^k|\phi\rangle = \cos(a + 2k\alpha)|\psi_0\rangle + \sin(a + 2k\alpha)|\psi_1\rangle. \tag{9}$$

Thus, consecutive iterations of amplitude amplification gradually shift the distribution of amplitudes. We must apply the right number of iterations so that the resulting state is as close to $|\psi_1\rangle$ as possible; this is achieved when $a + 2k\alpha \approx \frac{\pi}{2}$.

The standard amplitude amplification algorithm executes a total of $\lfloor \frac{\pi}{4\alpha} \rfloor$ consecutive iterations Q_χ starting on the initial state $|\psi\rangle$, then applies a measurement on the computational basis. The probability of finding a state in $\mathcal{H}_1$ is:

$$p_1 = \sin^2(\alpha + 2\lfloor \tfrac{\pi}{4\alpha} \rfloor \alpha) \geq \cos^2(2\alpha) \tag{10}$$

Observe that $p_1 = \cos^2(2\alpha) \approx 1$ for small α. Executing more (or less) than $\lfloor \frac{\pi}{4\alpha} \rfloor$ iterations will reduce the probability of success. In the unlikely event that the measurement does not find a state in $\mathcal{H}_1$, the whole algorithm is repeated again. The expected number of iterations is $\lfloor \frac{\pi}{4\alpha} \rfloor$, and each one applies O_χ once, meaning that the expected number of oracle calls is $\mathcal{O}\left(1/\sqrt{\rho}\right)$. Thus, according to Definition 1.1, the standard amplitude amplification algorithm is efficient.

1.3 Weak measurements

From a theoretical standpoint, the simplest kind of measurement is a *projection valued measure* (PVM): a set of orthogonal projections $\{P_i\}_{i\in\mathcal{I}}$ acting on the Hilbert space $\mathcal{H}$ that are mutually orthogonal and sum to the identity:

$$i \neq j \implies P_i P_j = 0 \qquad\qquad \sum_{i\in\mathcal{I}} P_i = I_{\mathcal{H}} \tag{11}$$

Equivalently, a PVM is determined by a set of closed subspaces $\mathcal{H}_i$, that are mutually orthogonal and span $\mathcal{H}$:

$$i \neq j \implies \mathcal{H}_i \perp \mathcal{H}_j \qquad\qquad \mathcal{H} = \bigoplus_{i\in\mathcal{I}} \mathcal{H}_i \tag{12}$$

We can switch between these descriptions by letting P_i be the orthogonal projection onto $\mathcal{H}_i$, and conversely letting $\mathcal{H}_i$ be the range of P_i.

Performing a PVM when the system is in state $|\psi\rangle \in \mathcal{H}$ results in outcome $i \in \mathcal{I}$ with probability

$$p_i = \langle\psi|P_i|\psi\rangle. \tag{13}$$

The measurement affects the state by projecting it onto $\mathcal{H}_i$ and renormalising:

$$\tfrac{1}{\sqrt{p_i}} P_i|\psi\rangle. \tag{14}$$

Amplitude amplification is interested in whether the state is in the marked subspace $\mathcal{H}_1$ or not. Often, this is checked by applying a measurement on the computational basis, and using the characteristic function χ (see Definition 1.1) to check if the measurement outcome is a marked element. Alternatively, we may implement a quantum controlled operation Λ_χ that acts on an extended space $\mathcal{H} \otimes \mathcal{P}$, where $\mathcal{P} = \text{span}\{0, 1\}$ is known as the *probe*. On the basis B, Λ_χ acts as

$$\Lambda_\chi |b, q\rangle = \begin{cases} |b, q\rangle & if b \in B \\ |b, q \oplus 1\rangle & if b \notin B \end{cases} \tag{15}$$

and may act on states in superposition. Then, given any state $\psi \in \mathcal{H}$, applying a PVM $\{I_\mathcal{H} \otimes |0\rangle\langle 0|, I_\mathcal{H} \otimes |1\rangle\langle 1|\}$ on $\Lambda_\chi|\psi, 0\rangle$ we know whether the outcome state is within $\mathcal{H}_0$ or within $\mathcal{H}_1$, without destroying all superposition. However, if this $\{\mathcal{H}_0, \mathcal{H}_1\}$ PVM is applied on every iteration, the state is always forced into one of the two subspaces, preventing the gradual evolution described in Section 1.2 and, hence, losing the quantum speed-up. Quantum feedback control [3, 9] has a similar problem. One solution is to instead use a *weak measurement*: "this is measurement which gives very little information about the system on average, but also disturbs the state very little" [5].

Neumark's dilation theorem [6] states that any valid notion of measurement on $\mathcal{H}$ comes from PVMs on a larger space $\mathcal{H} \otimes \mathcal{P}$. In this context, we refer to the space $\mathcal{P}$ as the *probe*, and its initial state is known. To perform a weak measurement, first entangle the original system $\mathcal{H}$ with the probe by applying a unitary operation E to $\mathcal{H} \otimes \mathcal{P}$, and then apply a PVM of the form $\{I_\mathcal{H} \otimes |i\rangle\langle i|\}_{i \in B(\mathcal{P})}$ indexed by an orthonormal basis $B(\mathcal{P})$ of $\mathcal{P}$. Without entanglement, this PVM would not provide any information about the state in $\mathcal{H}$, nor would it disturb it. Entanglement correlates the outcome of this PVM with the actual state in $\mathcal{H}$; the stronger this correlation, the more we learn about the state we wish to measure, but also the more we disturb it. In a weak measurement, the unitary E is chosen to make this correlation weak. Section 2.1 studies in detail the effect of the particular weak measurement our algorithm uses.

Weak measurements are a natural procedure in practice [1, 19, 8]: we can identify the probe $\mathcal{P}$ with the measuring device itself, and let the unitary operation $E = e^{-iHt}$ be driven by a Hamiltonian H describing the interaction of $\mathcal{H}$ and $\mathcal{P}$ during time t. By choosing an appropriately small value of t, we may make the interaction as weak as needed.

The field of quantum feedback control monitors a state via weak measurements (often a continuous measurement) and the stream of measurement outcomes is used to control the strength of a Hamiltonian that corrects the system; see [21] for a survey of the field. In the present paper we will work in the discrete-time regime: the weak measurement is applied at the end of every iteration, instead of continuously throughout the algorithm. One of the first examples of discrete-time quantum feedback appears in [3], where it was used to protect a qubit from decoherence. The concept was later experimentally realised in [9].

Weak measurements are rarely used for algorithmic purposes. The single application we are aware of is an implementation of Shor's algorithm using weak measurements instead of PVMs [14]. However, in that case, weak measurements are not used to monitor the quantum state throughout its evolution, but rather to statistically reconstruct the most probable outcome of the conventional PVM at the end of the computation.

1.4 While loops in quantum computing

There are multiple examples of quantum programming languages in the literature (see Refs. [10, 2, 16] for a few examples), and many have been shown to be expressive enough to describe quantum algorithms of practical relevance. Most of these quantum programming languages follow the slogan "quantum data, classical control" coined by Selinger [18], where algorithms are defined by providing quantum subroutines – described by unitary matrices – organised in a classical control flow. Via the control flow, the programmer must indicate which subroutine should be applied at each point of the algorithm, and this often depends on the outcome of a measurement.

In this paper, we are interested in while loops whose termination condition is given by the outcome of a measurement. A simple example of such a loop is:

```
while_{PVM} $q = |\perp\rangle$ do
  $U[\phi, q]$
end
```

where ϕ and q are variables representing quantum registers with state space $\mathcal{H}$ and $\mathcal{P}$ respectively. Given a selected state $|\perp\rangle \in \mathcal{P}$, this program first applies a PVM $\{P_\perp, P_\top\}$ with

$$P_\perp = I_\mathcal{H} \otimes |\perp\rangle\langle\perp| \tag{16}$$
$$P_\top = I_\mathcal{H} \otimes (I_\mathcal{P} - |\perp\rangle\langle\perp|) \tag{17}$$

and, as long as the measurement outcome is $\perp$, keeps applying unitary $U\colon \mathcal{H} \otimes \mathcal{P} \to \mathcal{H} \otimes \mathcal{P}$ on both registers (possibly entangling them). Once the measurement outcome is $\top$, the program terminates. This pseudocode notation is derived from previous work studying divergence of these kinds of loops [20].

2 While loop approach

This section presents an amplitude amplification algorithm that makes use of a while loop controlled by weak measurements. Take χ, $|\psi\rangle$, and $\mathcal{H} = \mathcal{H}_0 \oplus \mathcal{H}_1$ as in Definition 1.1, and Q_χ as in (5). Let ϕ be a variable with state space $\mathcal{H}$, and let q be the probe's qubit variable, with state space $\mathcal{P} \cong \mathbb{C}^2$ and chosen orthonormal basis $\{|\perp\rangle, |\top\rangle\}$. We write $\text{PVM}_\chi[\phi]$ to indicate that the PVM $\{\mathcal{H}_0, \mathcal{H}_1\}$ is applied

```
input: $\chi$, $|\psi\rangle$
output: $m$
begin
  $\phi \leftarrow |\psi\rangle$
  $q \leftarrow |\bot\rangle$
  while$_{PVM}$ $q = |\bot\rangle$ do
    $Q_\chi[\phi]$
    $E_\kappa[\phi \otimes q]$
  end
  $m \leftarrow \mathrm{PVM}_\chi[\phi]$
end
```

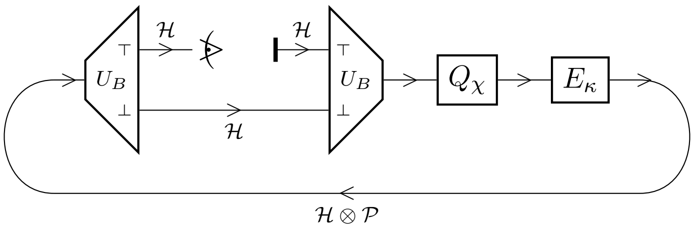

Figure 1: *Left:* pseudocode describing our algorithm. The while loop is controlled by a PVM acting only on the probe variable q; due to E_κ, this becomes a weak measurement of ϕ. *Right:* algorithm's information flow; U_B applies the canonical isomorphism $\mathcal{H} \otimes \mathcal{P} \cong \mathcal{H} \oplus \mathcal{H}$, separating the orthogonal cases $|\bot\rangle$ and $|\top\rangle$.

to the state of variable ϕ. Our amplitude amplification algorithm is described by the pseudocode shown in Figure 1, using the notation described in Section 1.4 and where

$$E_\kappa = P_0 \otimes I_{\mathcal{P}} + P_1 \otimes R_\kappa \tag{18}$$

$$R_\kappa = \begin{pmatrix} \sqrt{1-\kappa} & \sqrt{\kappa} \\ \sqrt{\kappa} & -\sqrt{1-\kappa} \end{pmatrix} \tag{19}$$

for some $\kappa \in [0,1]$. Additionally, Figure 1 gives an informal transcription of the pseudocode into the discrete-time quantum walk model, providing some intuition of the algorithm's information flow. Notice that E_κ is an operation that acts differently on $\mathcal{H}_0$ and $\mathcal{H}_1$ thus, similarly to Λ_χ (see Section 1.3) it internally queries the oracle function χ, and so does Q_χ.

The rest of this section is structured as follows. In Section 2.1 we will study, for any iteration n, the state $|\phi\rangle^{(n)}$ held by variable ϕ. Similarly to the situation in Section 1.2, the state $|\phi\rangle^{(n)}$ is a superposition of the form

$$|\phi^{(n)}\rangle = \cos a_n |\psi_0\rangle + \sin a_n |\psi_1\rangle \tag{20}$$

where $|\psi_1\rangle = P_1|\psi\rangle$ is in the marked subspace $\mathcal{H}_1$ and $|\psi_0\rangle = P_0|\psi\rangle$ is in $\mathcal{H}_0$. We give the evolution of the angle a_n across iterations, which comes from the angle of the previous iteration, adding the contribution of Q_χ and subtracting θ_n; the collapsing effect of the measurement:

$$\begin{aligned} a_{n+1} &= a_n + 2\alpha - \theta_n \\ a_0 &= \alpha \approx \arcsin\sqrt{\rho}. \end{aligned} \tag{21}$$

In Proposition 2.3 we prove that, for weak enough measurements $\kappa \leq \sqrt{\rho}$, $|\theta_n| < \alpha$. This means that, for any iteration n, $a_{n+1} \geq a_n + \alpha$ and thus the state develops towards $|\psi_1\rangle$ (i.e. to $a \approx \frac{\pi}{2}$) at a steady pace. Finally, Section 2.2 shows that, within $\mathcal{O}(1/\kappa)$ iterations, the quantum state stays close to $|\psi_1\rangle$ for long enough so that, with high probability, one of the weak measurements may detect it, terminating the algorithm with success. We will set $\kappa \approx \sqrt{\rho}$ so that, according to Definition 1.1, our algorithm is efficient.

There is a trade-off between the strength of the weak measurement κ and the expected number of iterations. If $\kappa \gg \sqrt{\rho}$, there will be too much collapse on each iteration, causing the algorithm to behave classically and lose the quantum speed-up. But, if $\kappa \ll \sqrt{\rho}$ the complexity of the algorithm $\mathcal{O}(1/\kappa)$ will increase. Thus, in contrast to the standard amplitude amplification, where the number of iterations needs to be chosen a priori to be $N \approx \frac{\pi}{4\sqrt{\rho}}$, we do not indicate for how long our algorithm runs, but instead need to fix the strength of the measurement to be $\kappa \approx \sqrt{\rho}$.

2.1 Geometry of a single iteration

In Section 1.2 we used the fact that the state $|\phi\rangle \in \mathcal{H}$ at any point of the execution was described by a superposition of the form

$$|\phi\rangle = \cos a|\psi_0\rangle + \sin a|\psi_1\rangle \tag{22}$$

for some angle $a \in [0, 2\pi)$. This allowed us to describe the evolution of the state in terms of the angle a. We now study what is the effect of the weak measurement in the algorithm from Figure 1.

Proposition 2.1. *For a state $|\phi^{(n)}\rangle$ of the form* (22)*, the weak measurement consisting of first applying E_κ and then PVM $\{I_{\mathcal{H}} \otimes P_\perp, I_{\mathcal{H}} \otimes P_\top\}$ to $|\phi^{(n)}\rangle|\perp\rangle$ returns*

$$|\phi_\top^{(n)}\rangle = |\psi_1\rangle \tag{23}$$

when the outcome is $\top$, which occurs with probability $p_\top = \kappa \sin^2 a_n$, and

$$|\phi_\perp^{(n)}\rangle = \frac{1}{\sqrt{p_\perp}}\left(\cos a_n|\psi_0\rangle + \sqrt{1-\kappa}\sin a_n|\psi_1\rangle\right) \tag{24}$$

when the outcome is $\perp$, which occurs with probability $p_\perp = 1 - p_\top$.

Proof. Applying E_κ to $|\phi^{(n)}\rangle|\perp\rangle$ results in an entangled state

$$\begin{aligned} E_\kappa|\phi^{(n)}\rangle|\perp\rangle &= \cos a_n|\psi_0\rangle|\perp\rangle + \sin a_n|\psi_1\rangle\left(\sqrt{1-\kappa}|\perp\rangle + \sqrt{\kappa}|\top\rangle\right) \\ &= \sqrt{p_\perp}\,|\phi_\perp^{(n)}\rangle\,|\perp\rangle + \sqrt{p_\top}\,|\phi_\top^{(n)}\rangle\,|\top\rangle. \end{aligned} \tag{25}$$

Applying each of the PVM's projectors yields the results claimed. □

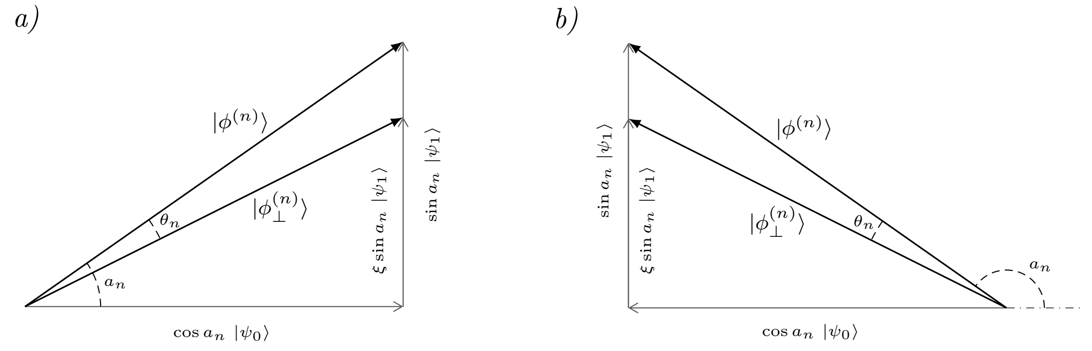
a)
$|\phi^{(n)}\rangle$
$|\phi_\perp^{(n)}\rangle$
θ_n
a_n
$\xi \sin a_n\ |\psi_1\rangle$
$\sin a_n\ |\psi_1\rangle$
$\cos a_n\ |\psi_0\rangle$
b)
$|\phi^{(n)}\rangle$
$|\phi_\perp^{(n)}\rangle$
θ_n
a_n
$\sin a_n\ |\psi_1\rangle$
$\xi \sin a_n\ |\psi_1\rangle$
$\cos a_n\ |\psi_0\rangle$

Figure 2: Geometric relation between the angle a_n before weak measurement and the offset θ_n after measuring outcome $\perp$. The construction is provided when *a)* a_n is in the first quadrant and when *b)* a_n is in the second quadrant. $\xi = \sqrt{1-\kappa}$.

Proposition 2.1 determines the probability $p_\top$ that the while loop terminates on the n-th iteration, where a_n is the angle representing the state. Thanks to $|\phi_\top^{(n)}\rangle$ being equal to $|\psi_1\rangle \in \mathcal{H}_1$, we know that the algorithm always produces a marked element upon termination. Otherwise, on outcome $\perp$ another iteration of the while loop occurs, now with variable ϕ holding the state $|\phi_\perp^{(n)}\rangle$ and the probe holding state $|\perp\rangle$. Furthermore, Proposition 2.1 shows that $|\phi_\perp^{(n)}\rangle$ is still a superposition of $|\psi_0\rangle$ and $|\psi_1\rangle$; we now wish to find an angle a'_n that describes the superposition in the form (22). Figure 2 shows the relationship between a_n and a'_n in terms of an offset angle θ_n. Then, Q_χ will update the angle, leading to the following recurrence equation describing the evolution of the state throughout the iterations:

$$\begin{aligned} a_{n+1} &= a_n - \theta_n + 2\alpha \\ a_0 &= \alpha \approx \arcsin\sqrt{\rho}. \end{aligned} \tag{26}$$

Let's find an algebraic expression for θ_n in terms of a_n and κ. To reduce clutter, we will use the shorthand

$$\xi = \sqrt{1-\kappa} \tag{27}$$

Suppose $a_n \in [0, \frac{\pi}{2}]$ and use the properties of sines on the triangle in Figure 2a to obtain:

$$\frac{\sin(a_n - \theta_n)}{\xi \sin a_n} = \frac{\sin(\pi/2 - a_n + \theta_n)}{\cos a_n} \tag{28}$$

This simplifies as follows:

$$\begin{aligned} & \frac{\sin(a_n-\theta_n)}{\xi\sin a_n} = \frac{\cos(a_n-\theta_n)}{\cos a_n} \\ \iff \quad & \tan(a_n-\theta_n) = \xi\tan a_n \qquad (29) \\ \iff \quad & \frac{\tan a_n - \tan\theta_n}{1+\tan a_n\tan\theta_n} = \xi\tan a_n \end{aligned}$$

Solving for θ_n gives:

$$\theta_n = \arctan\left(\frac{(1-\xi)\tan a_n}{1+\xi\tan^2 a_n}\right) \tag{30}$$

If $a_n \in \left[\frac{\pi}{2},\pi\right]$ instead, a similar analysis yields:

$$\theta_n = -\arctan\left(\frac{(1-\xi)\tan a_n}{1+\xi\tan^2 a_n}\right) \tag{31}$$

which only differs from equation (30) in the sign. If a_n is in the third quadrant, then we obtain equation (30) and, if a_n is in the fourth quadrant, we get (31).

Remark 2.2. *These sign changes are convenient: the geometric analysis in Figure 2 shows that in the first (and third) quadrant, the angle a'_n is $a_n - |\theta_n|$, whereas in the second (and fourth) quadrant it is $a_n + |\theta_n|$. Succinctly, $a'_n = a_n - \theta_n$, independently of which quadrant a_n is in.*

Proposition 2.3. *The following bounds apply to θ_n:*

$$\begin{array}{lcl} 0 \le \theta_n \le a_0 & \textit{if} & a_n \in [0,\frac{\pi}{2}] \cup [\pi,\frac{3\pi}{2}] \\ -a_0 \le \theta_n \le 0 & \textit{if} & a_n \in [\frac{\pi}{2},\pi] \cup [\frac{3\pi}{2},2\pi] \end{array} \tag{32}$$

if and only if

$$\kappa \le \frac{4\sqrt{\rho}}{(1+\sqrt{\rho})^2}. \tag{33}$$

Proof. The geometric construction of θ_n in Figure 2 makes it clear that $0 \le |\theta_n| \le \frac{\pi}{2}$ in any case. This gives the desired lower bound when a_n is in the first or third quadrant, and the desired upper bound when a_n is in the second or fourth quadrants.

When a_n is in the first quadrant we already know $\theta_n \le \frac{\pi}{2}$ but we want a tighter upper bound. To do so, regard (30) as a function of a_n and study its maximum value. First we find the critical points, where the derivative

$$\frac{\mathrm{d}\theta_n}{\mathrm{d}a_n} = 1 - \frac{\xi}{\cos^2 a_n + \xi^2\sin^2 a_n} \tag{34}$$

vanishes. Within $a_n \in [0, \frac{\pi}{2}]$ this happens at:

$$a_n = \arccos\left(\sqrt{\frac{\xi}{\xi+1}}\right). \tag{35}$$

Applied to equation (30) this gives a tight upper bound:

$$\theta_n \leq \arctan\left(\sqrt{\frac{(1-\xi)^2}{4\xi}}\right) \tag{36}$$

We wish to find what κ needs to be so that $a_0 \geq \theta_n$ is satisfied. Thanks to $a_0, \theta_n \in [0, \frac{\pi}{2}]$, $a_0 \geq \theta_n$ is true if and only if $\sin a_0 \geq \sin \theta_n$. Using (36), the equality $\sin(\arctan(x)) = \frac{x}{\sqrt{x^2+1}}$ and some basic algebra we find that $a_0 \geq \theta_n$ is true if and only if

$$\sin a_0 \geq \frac{1-\xi}{1+\xi}. \tag{37}$$

By definition (27), we know that $\xi = \sqrt{1-\kappa}$ and (21) implies that $\sqrt{\rho} \approx \sin a_0$. Using these identities on (37) and solving for κ we conclude that $a_0 \geq \theta_n$ if and only if

$$\kappa \leq \frac{4\sqrt{\rho}}{(1+\sqrt{\rho})^2} \tag{38}$$

thus, proving the claim when a_n is in the first quadrant. The same argument establishes the upper bound when a_n is in the third quadrant. When a_n is in either the second or fourth quadrants we must take into account that equation (30) yields a negative value. In those cases, a similar argument yields the lower bound $-a_0 \leq \theta_n$ for the same condition on κ. □

The bounds given in Proposition 2.3 ensure that the sequence of angles $\{a_i\}_{i\in\mathbb{N}}$ is monotonically increasing, as stated in the following corollary.

Corollary 2.4. *For all $k, \ell \in \mathbb{N}$:*

$$a_k + \ell a_0 \;\leq\; a_{k+\ell} \;\leq\; a_k + 3\ell a_0. \tag{39}$$

if and only if κ satisfies (33).

Proof. From Proposition 2.3 and equation (21) it follows that

$$a_k + \ell(2a_0 - a_0) \;\leq\; a_{k+\ell} \;\leq\; a_k + \ell(2a_0 + a_0). \tag{40}$$

This is the inequality to be proven. □

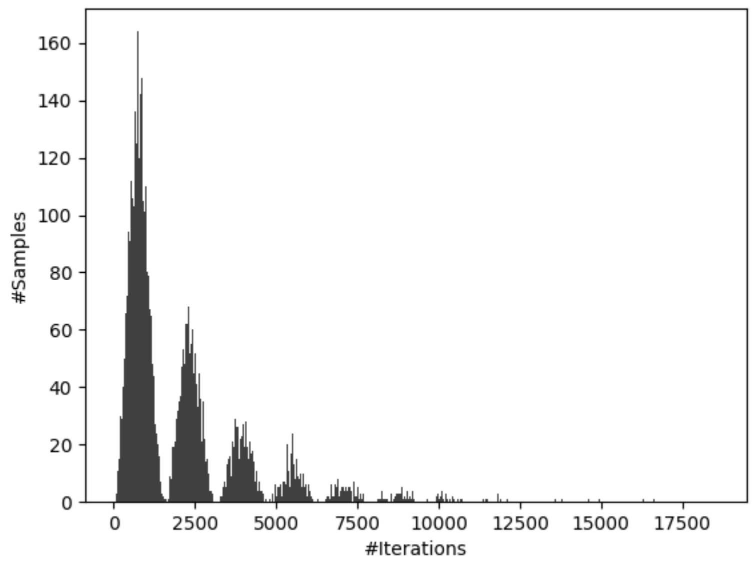

Figure 3: Histogram of the number of iterations before success (termination) of our algorithm. The histogram is composed of 10000 samples; these samples were obtained by sampling the success probability $\kappa \sin^2 a_n$ where a_n is given by equation (21). We used parameters $\rho = 10^{-6}$ and $\kappa = \sqrt{\rho}$.

2.2 Running time

According to Corollary 2.4, the angle a_n monotonically increases at a steady pace throughout the algorithm. Therefore, during some iterations the angle will be close to $\frac{\pi}{2}$ and thus the probability of the algorithm terminating will be at its maximum. The histogram of Figure 3 clearly displays this behaviour: the algorithm alternates between intervals of iterations where termination is likely (i.e. peaks) and intervals where the angle a_n is close to 0 or π and thus termination is unlikely (i.e. troughs). We refer to these two distinct intervals as *active* and *latent* iterations, respectively.

In this section we formally define these active and latent iterations and estimate their proportion across the algorithm. Then we calculate the number of active iterations required for the algorithm to succeed with probability over $\frac{1}{2}$, and conclude that $\mathcal{O}\left(1/\sqrt{\rho}\right)$ many calls to the oracle are enough for the success probability to get arbitrarily close to one.

Definition 2.5. *For any $m \in \mathbb{N}$, the m-th iteration of the algorithm is said to be an* active iteration *if and only if*

$$\exists i \in \mathbb{N}: \qquad \frac{\pi}{4} + i\pi \;\leq\; a_m \;\leq\; \frac{3\pi}{4} + i\pi. \tag{41}$$

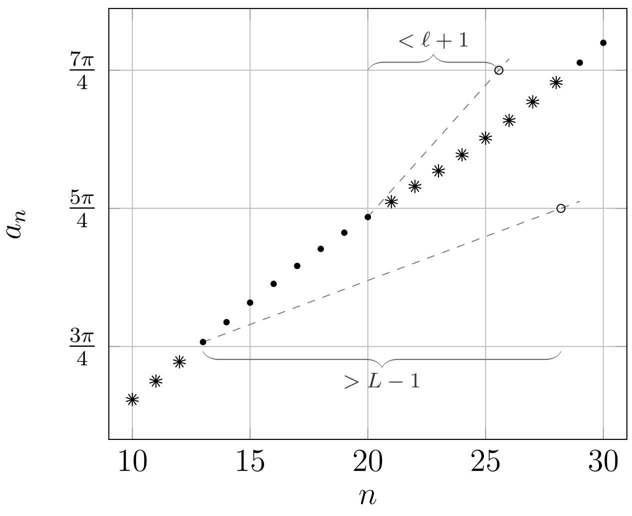

Figure 4: Evolution of the angle across iterations 10 to 30, for $\rho = \frac{1}{100}$. The black dots indicate the value of a_n at each iteration, starred dots correspond to active iterations. The angle increases by $2a_0$ on each iteration, with some small disturbance introduced by θ_n. The dashed lines represent the lower and upper bounds of a_n, used to estimate ℓ and L in the general case. In this case, $L = 8 = \ell$.

Otherwise, it is called a latent iteration*.*

Figure 4 shows the evolution of the angle across iterations for $\rho = \frac{1}{100}$. Only iterations from 10 to 30 are displayed, but because n does not appear as a term in either equation (21) or (30), the behaviour shown may be extrapolated to any interval of iterations. We may use the bounds from Corollary 2.4 to give an upper bound on the number L of consecutive latent iterations.

Proposition 2.6. *If $a_k > \frac{3\pi}{4} + i\pi$ and $a_{k'} < \frac{5\pi}{4} + i\pi$ for some $i \in \mathbb{N}$ and $k' > k$, then $k' - k < \frac{\pi}{2a_0}$.*

Proof. Using the lower bound from Corollary 2.4 and the assumption $a_k > \frac{3\pi}{4} + i\pi$:

$$\begin{aligned} a_{k'} &= a_{k+(k'-k)} \\ &\geq a_k + (k'-k)a_0 \\ &> \tfrac{3\pi}{4} + i\pi + (k'-k)a_0 \end{aligned} \tag{42}$$

Hence:

$$\tfrac{5\pi}{4} + i\pi > a_{k'} > \tfrac{3\pi}{4} + i\pi + (k'-k)a_0 \tag{43}$$

But this simplifies to $k' - k < \frac{\pi}{2a_0}$. □

If k is the first iteration in an interval of latent ones and k' is the last, then $k' = k + (L - 1)$. The previous proposition ensures that $L < \frac{\pi}{2a_0} + 1$. Similarly, we can estimate the number ℓ of consecutive active iterations.

Proposition 2.7. *If $a_k < \frac{\pi}{4} + i\pi$ and $a_{k'} > \frac{3\pi}{4} + i\pi$ for some $i \in \mathbb{N}$ and $k' > k$, then $k' - k > \frac{\pi}{6a_0}$.*

Proof. Using the upper bound from Corollary 2.4 and the assumption $a_k < \frac{\pi}{4} + i\pi$:

$$\begin{aligned} a_{k'} &= a_{k+(k'-k)} \\ &\leq a_k + 3(k'-k)a_0 \\ &< \tfrac{\pi}{4} + i\pi + 3(k'-k)a_0 \end{aligned} \tag{44}$$

Hence:

$$\tfrac{3\pi}{4} + i\pi < a_{k'} < \tfrac{\pi}{4} + i\pi + 3(k'-k)a_0 \tag{45}$$

But this simplifies to $k' - k > \frac{\pi}{6a_0}$. □

If k is the last latent iteration before an active interval and k' the first latent one after it, then $k' = k + (\ell + 1)$, and previous proposition ensures that $\ell > \frac{\pi}{6a_0} - 1$. The dashed lines in Figure 4 summarise these bounds. Now we can estimate the proportion of active iterations across the algorithm.

Proposition 2.8. *The proportion γ of active iterations across the algorithm is:*

$$\gamma > \frac{1}{4} - \frac{3a_0}{2\pi}. \tag{46}$$

Proof. Due to Corollary 2.4 we know that a_n increases with n, and thus intervals of active and latent iterations will alternate throughout the runtime of the algorithm. We further know an upper bound for the length of any latent interval, $L < \frac{\pi}{2a_0} + 1$, and a lower bound for the length of any active interval $\ell > \frac{\pi}{6a_0} - 1$. By definition,

$$\gamma = \frac{\ell}{L + \ell}, \tag{47}$$

or equivalently,

$$\gamma^{-1} = \frac{L}{\ell} + 1. \tag{48}$$

The latter expression is more amenable to applying our bounds for L and ℓ. With these bounds and some basic algebra it follows that

$$\gamma^{-1} < \frac{4\pi}{\pi - 6a_0}, \tag{49}$$

which implies the claim. □

Remark 2.9. *When ρ (and thus a_0) is small, the proportion of active iterations given in Proposition 2.8 is approximately $\frac{1}{4}$. This proportion was estimated using the arguably loose bounds from Corollary 2.4, which were chosen for the sake of simplicity. Tighter bounds would yield a better estimate of the proportion of active iterations which, as shown in Figure 4, appears to be close to $\frac{1}{2}$ when ρ is small.*

Next, we must calculate the number of active iterations required to achieve a success probability larger than $\frac{1}{2}$. As stated in Proposition 2.1, the algorithm terminates on the n-th iteration with probability

$$p_\top = \kappa \sin^2 a_n \tag{50}$$

and it is always successful upon termination. For any active iteration m:

$$p_\top = \kappa \sin^2 a_m \;\geq\; \kappa \sin^2(\tfrac{\pi}{4}) = \frac{\kappa}{2} \tag{51}$$

where we have used the definition of active iteration (41). The total probability P_{succ} of the algorithm succeeding at some point within $\frac{2}{\kappa}$ active iterations is:

$$P_{\text{succ}} \geq 1 - \left(1 - \tfrac{\kappa}{2}\right)^{\frac{2}{\kappa}} \tag{52}$$

If $x \in (0,1)$ then $(1-x)^{\frac{1}{x}} < \frac{1}{e}$. Therefore:

$$P_{\text{succ}} \geq 1 - \tfrac{1}{e} \geq \tfrac{1}{2} \tag{53}$$

if the algorithm is allowed to run for $\frac{2}{\kappa}$ active iterations or more.

It only remains to calculate how many iterations are needed to be certain that at least $\frac{2}{\kappa}$ active iterations occur. Using the proportion of active iterations derived in Proposition 2.8 we know that the total number T of iterations required to reach a probability of success greater than $\frac{1}{2}$ is

$$T = \frac{2}{\kappa}\gamma^{-1} < \frac{8\pi}{\kappa(\pi - 6a_0)}. \tag{54}$$

Assuming ρ is small, $a_0 \approx \arcsin\sqrt{\rho}$ will also be, and then T will be bounded from above by $\frac{8}{\kappa}$.[1]

$$T < \frac{8\pi}{\sqrt{\rho}(\pi - 6\arcsin\sqrt{\rho})} \tag{55}$$

If we were unlucky and the algorithm does not succeed after T iterations, it will continue iterating, replicating the same behaviour for another T iterations or succeeding (and thus halting) in the process. For any $c \in \mathbb{N}$, the probability of

[1]In fact, Remark 2.9 argues that the required number of iterations is closer to $\frac{4}{\sqrt{\rho}}$.

success after cT iterations is greater than $1-(1-\frac{1}{2})^c$, *i.e.* the probability of success increases exponentially. Therefore, we can achieve an arbitrarily high success probability within $\mathcal{O}(1/\kappa)$ many iterations. Notice that each iteration calls the oracle twice: once when checking the while-loop condition, and a second time within the implementation of E_κ, contributing to the oracle-complexity of the algorithm by a factor of two. Finally, we set $\kappa = \sqrt{\rho}$, which satisfies the inequality (33) required by Proposition 2.3. Overall, the expected number of oracle calls stays within $\mathcal{O}\left(1/\sqrt{\rho}\right)$, and the algorithm is efficient according to Definition 1.1.

3 Comparison to test-restart loop

In this section, we compare the algorithm described in Section 2 with an alternative algorithm that makes does not make use of weak-measurements, but instead runs for an arbitrary number of iterations and then applies a PVM, restarting if the outcome is not a marked element. We find that this alternative *test-restart* approach has essentially the same statistical performance as our *weakly-measured* approach.

Let N_{wm} be the random variable denoting the total number of iterations that may occur when running the algorithm in Figure 1. Figure 3 shows a histogram of N_{wm}. Considering that the algorithm terminates upon (first) success, it is natural that the histogram follows the behaviour of a geometric distribution: larger values of N_{wm} are less likely because they imply longer sequences of consecutive failures.

In contrast to the approach we proposed in Figure 1, we now consider the following *test-restart* approach

```
input: $\chi$, $|\psi\rangle$
output: $m$
begin
  $m \leftarrow \mathrm{PVM}_\chi \phi$
  while $\chi(m) = 0$ do
    $\phi \leftarrow |\psi\rangle$
    $r \leftarrow 1$
    while $\sqrt{\rho} < r$ do
      $r \leftarrow$ rnd(0,1)
      $Q_\chi[\phi]$
    $m \leftarrow \mathrm{PVM}_\chi[\phi]$
  end
end
```

where rnd(0,1) picks a random number from the real interval $[0,1]$. This alternative algorithm does not use weak measurements. The inner while loop applies Q_χ a random number of times described by a geometric distribution with parameter $\sqrt{\rho}$. The state is measured with a PVM only after exiting the inner loop and, if we fail to find a marked element, the whole process is repeated again, setting the state to its initial value $|\psi\rangle$. We use the random variable N_{tr} to denote the total number

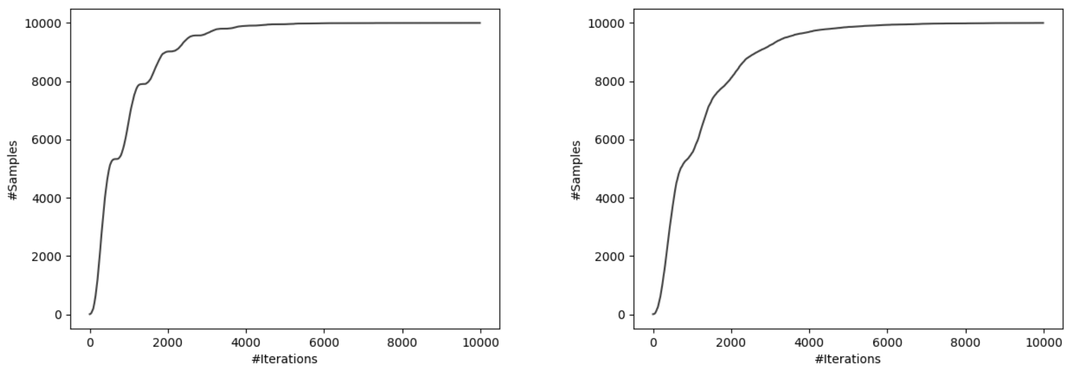

Figure 5: Cumulative distribution of N_{wm} (left) and N_{tr} (right) from a set of 10000 samples each. We used parameters $\rho = 10^{-6}$ and $\kappa = \sqrt{\rho}$. Both distributions have a similar curve, although N_{wm} displays a staggered growth due to the alternation between active and latent iterations (see Section 2.2).

of applications of Q_χ before success. Unlike in the algorithm from Figure 1, there is no superposition of control flow: these while loops are purely classical.

It is natural that the expected value of N_{tr} is $1/\sqrt{\rho}$: the expected value of consecutive applications of Q_χ is $\frac{1}{\sqrt{\rho}}$, and in that case the PVM finds a marked state with high probability. However, one might expect that the stochastic behaviour of N_{tr} would be noticeably more erratic than that of N_{wm} as, in the test-restart case, failing to measure a marked state results in having to start from scratch. Interestingly, Figure 5 illustrates that this is not the case:

$$P(N_{tr} \leq n) \approx P(N_{wm} \leq n) \tag{56}$$

for any $n \in \mathbb{N}$, meaning that both distributions have roughly the same median and percentiles. These values are relevant for worst-case analysis, as low and high percentiles describe the frequency at which extreme values of N_{tr} and N_{wm} may occur. Moreover, both distributions have essentially the same variance.

Notice, however, that the probability distributions of N_{tr} and N_{wm} are not exactly the same.[2] This is made evident in Figure 5, where the cumulative distribution of N_{wm} goes through periodic non-increasing intervals due to latent iterations, whereas that of N_{tr} is smoother in comparison.

In summary, although the test-restart and weakly-measured approaches are fundamentally different, both provide the same average-case and worst-case com-

[2]We performed statistical tests on the set of samples from N_{tr} and N_{wm} and, with sample sizes of 10000 and significance level of $\alpha = 0.01$ (i.e. 99% confidence), both the Kolmogorov-Smirnov test [12] and the Anderson-Darling test [17] indicate the distributions are different.

plexity. This means that weak measurements do not yield any algorithmic advantage in this scenario. As further work, we intend to study whether there is an algorithmic application of weak measurements where a test-restart loop would perform poorly.

4 Discussion

In the standard amplitude amplification approach (Section 1.2), a PVM is applied after a particular number of iterations, so implementing it requires the ability to control the hardware well enough to count iterations and apply the projective measurement at the right time. Experimentalists may find it easier to build hardware where a certain quantum subsystem is (weakly) measured on every iteration.

In this paper, we have reproduced Mizel's [15] version of amplitude amplification and analysed its evolution using simple geometry. The algorithm is parametrised by κ, which characterises how much of the state is collapsed by the weak measurement. In the standard amplitude amplification algorithm, it is necessary to tune the number of iterations according to the value of ρ; in a similar way, we have shown that our algorithm is efficient only if $\kappa \approx \sqrt{\rho}$, i.e. when the measurements applied are weak enough.

Extensions of the standard amplitude amplification algorithm where the initial probability ρ is unknown have been considered in the literature [4]. These proposals consist in running the standard algorithm multiple times, assuming different values of ρ – gradually decreased by a factor of two each time – until the actual value of ρ is reached and the algorithm succeeds. These approaches are efficient and may be straightforwardly applied to our version of the algorithm. However, they require restarting the algorithm multiple times, and thus they are not a good match to the moral behind our perspective: gaining information without destroying the state. A possible direction of future work is to exploit weak measurements to guide the search towards the value of ρ when unknown.

Our algorithm applies a weak measurement at the end of each iteration, at discrete times. It would be interesting to study whether it is possible to *continuously* apply a weak measurement on the probe space $\mathcal{P}$ and achieve the same behaviour. If it is possible, then this algorithm would have a simple implementation in the laboratory: evolve the system $\mathcal{H} \otimes \mathcal{P}$ while a passive device is constantly measuring $\mathcal{P}$, thus avoiding the need of an active control mechanism. The main obstacle of this continuous approach may be the quantum Zeno effect. This idea was not discussed by Mizel in [15] and, as far as we know, it has not been considered in the literature yet. We intend to study this problem in future work, using the formalism of quantum trajectories for continuous measurements [13].

References

[1] Y. Aharonov and L. Vaidman. Properties of a quantum system during the time interval between two measurements. *Phys. Rev. A*, 41:11–20, Jan 1990.

[2] S. Bettelli, T. Calarco, and L. Serafini. Toward an architecture for quantum programming. *The European Physical Journal D-Atomic, Molecular, Optical and Plasma Physics*, 25(2):181–200, 2003.

[3] A. M. Brańczyk, P. E. M. F. Mendonça, A. Gilchrist, A. C. Doherty, and S. D. Bartlett. Quantum control of a single qubit. *Physical Review A*, 75(1):012329, 2007.

[4] G. Brassard, P. Hoyer, M. Mosca, and A. Tapp. Quantum amplitude amplification and estimation. *Contemporary Mathematics*, 305:53–74, 2002.

[5] T. A. Brun. A simple model of quantum trajectories. *American Journal of Physics*, 70(7):719–737, 2002.

[6] P. Busch, M. Grabowski, and P. Lahti. *Operational Quantum Physics.* Springer, 1995.

[7] M. Davis. *Computability and unsolvability.* Courier Corporation, 2013.

[8] J. Dressel, M. Malik, F. M. Miatto, A. N. Jordan, and R. W. Boyd. Colloquium: Understanding quantum weak values: Basics and applications. *Rev. Mod. Phys.*, 86:307–316, Mar 2014.

[9] G. G. Gillett, R. B. Dalton, B. P. Lanyon, M. P. Almeida, M. Barbieri, G. J. Pryde, J. L. O'Brien, K. J. Resch, S. D. Bartlett, and A. G. White. Experimental feedback control of quantum systems using weak measurements. *Physical review letters*, 104(8):080503, 2010.

[10] A. S. Green, P. LeFanu Lumsdaine, N. J. Ross, P. Selinger, and B. Valiron. Quipper: A scalable quantum programming language. In *Proceedings of the 34th ACM SIGPLAN Conference on Programming Language Design and Implementation*, PLDI '13, page 333–342, New York, NY, USA, 2013. Association for Computing Machinery.

[11] L. K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 212–219, 1996.

[12] J. L. Hodges. The significance probability of the smirnov two-sample test. *Arkiv för Matematik*, 3:469–486, 1958.

[13] K. Jacobs and D. A. Steck. A straightforward introduction to continuous quantum measurement. *Contemporary Physics*, 47(5):279–303, 2006.

[14] A. P. Lund. Efficient quantum computing with weak measurements. *New Journal of Physics*, 13(5):053024, may 2011.

[15] A. Mizel. Critically damped quantum search. *Phys. Rev. Lett.*, 102:150501, Apr 2009.

[16] J. W. Sanders and P. Zuliani. Quantum programming. In R. Backhouse and J. N. Oliveira, editors, *Mathematics of Program Construction*, pages 80–99, Berlin, Heidelberg, 2000. Springer Berlin Heidelberg.

[17] F. W. Scholz and M. A. Stephens. K-sample anderson–darling tests. *Journal of the American Statistical Association*, 82(399):918–924, 1987.

[18] P. Selinger. Towards a quantum programming language. *Mathematical Structures in Computer Science*, 14(4):527–586, 2004.

[19] L. Vaidman. Weak-measurement elements of reality. *Foundations of Physics*, 26(7):895–906, 1996.

[20] M. Ying and Y. Feng. Quantum loop programs. *Acta Informatica*, 47(4):221–250, 2010.

[21] J. Zhang, Y. x. Liu, R.-B. Wu, K. Jacobs, and F. Nori. Quantum feedback: theory, experiments, and applications. *Physics Reports*, 679:1–60, 2017.