



Differentiable Quantum Programming with Unbounded Loops

WANG FANG, Institute of Software, Chinese Academy of Sciences, China and University of Chinese Academy of Sciences, China

MINGSHENG YING, Institute of Software, Chinese Academy of Sciences, China and Tsinghua University, China

XIAODI WU, University of Maryland, United States

The emergence of variational quantum applications has led to the development of automatic differentiation techniques in quantum computing. Existing work has formulated differentiable quantum programming with bounded loops, providing a framework for scalable gradient calculation by quantum means for training quantum variational applications. However, promising parameterized quantum applications, e.g., quantum walk and unitary implementation, cannot be trained in the existing framework due to the natural involvement of unbounded loops. To fill in the gap, we provide the first differentiable quantum programming framework with unbounded loops, including a newly designed differentiation rule, code transformation, and their correctness proof. Technically, we introduce a randomized estimator for derivatives to deal with the infinite sum in the differentiation of unbounded loops, whose applicability in classical and probabilistic programming is also discussed. We implement our framework with Python and Q# and demonstrate a reasonable sample efficiency. Through extensive case studies, we showcase an exciting application of our framework in automatically identifying close-to-optimal parameters for several parameterized quantum applications.

CCS Concepts: • **Software and its engineering** → *General programming languages*; • **Mathematics of computing** → **Automatic differentiation**;

Additional Key Words and Phrases: Quantum programming languages, differentiable programming, quantum machine learning, unbounded loops

ACM Reference format:

Wang Fang, Mingsheng Ying, and Xiaodi Wu. 2023. Differentiable Quantum Programming with Unbounded Loops. *ACM Trans. Softw. Eng. Methodol.* 33, 1, Article 19 (November 2023), 63 pages.

<https://doi.org/10.1145/3617178>

W. Fang and M. Ying were partially supported by the National Key R&D Program of Chinander Grant No. 2018YFA0306701 and the National Natural Science Foundation of China under Grant No. 61832015. X. W. was partially funded by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, Quantum Testbed Pathfinder Program under Award Number DE-SC0019040 and by the U.S. National Science Foundation grant CCF-1942837 (CAREER). Authors' addresses: W. Fang, State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, the 4th Zhongguancun South Fourth Street, Haidian District, Beijing, 100190, China and University of Chinese Academy of Sciences, China; e-mail: fangw@ios.ac.cn; M. Ying, State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, the 4th Zhongguancun South Fourth Street, Haidian District, Beijing, 100190, China and Department of Computer Science and Technology, Tsinghua University, China; e-mail: yingms@ios.ac.cn; X. Wu, Department of Computer Science, University of Maryland, College Park, MD 20740 and Joint Center for Quantum Information and Computer Science, University of Maryland, College Park, MD 20740; e-mail: xwu@cs.umd.edu.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from [permissions@acm.org](https://permissions.acm.org).

© 2023 Copyright held by the owner/author(s). Publication rights licensed to ACM.

1049-331X/2023/11-ART19 \$15.00

<https://doi.org/10.1145/3617178>

1 INTRODUCTION

Inspired by the advantage of neural networks with program features (e.g., controls) over the plain ones [26, 27], the notion of *differentiable programming* has been introduced [1, 7, 25, 66] as a new programming paradigm, where programs become parameterized and differentiable, and has recently stimulated active investigation (e.g., References [34, 47, 48, 60]). Specifically, many efforts have been devoted to the development of *automatic differentiation* (e.g., References [20, 28]) for various program constructs.

Quantum programming is one specific type of programming that would benefit from the study of automatic differentiation. With the availability of the 50~100-qubit machines, near-term **Noisy Intermediate-Scale Quantum (NISQ)** machines [56] have become the major platform for quantum applications. Parameterized (or variational) quantum circuits, introduced as a quantum machine learning model with remarkable expressive power [10], are one compelling candidate of NISQ applications, including examples such as **variational quantum eigensolver (VQE)** [36, 54], quantum neural networks [9, 19, 24], to the **quantum approximate optimization algorithm (QAOA)** [22, 23, 32]. Similar to classical machine learning, gradient-based methods are employed to train the loss functions, which, however, now depend on the read-outs of quantum computation. Thus, the “quantum” gradient calculation has a similar complexity of simulating quantum circuits, which is infeasible for classical computation.

Automatic differentiation (AD) on quantum programs, which would enable the ability of computing quantum gradients efficiently by quantum computation, is thus critical for the scalability of variational quantum applications. However, it is *a priori* unclear whether the AD technique could extend to the quantum setting at all due to a few fundamental differences between quantum and classical. First, an appropriate formulation of the differentiation in quantum computing is important, because the outcomes of quantum programs are quantum states rather than classical variables. Second, the quantum no-cloning theorem [71] prohibits the duplication of intermediate states in quantum programs, which prohibits the natural extension of the classical forward-mode and reverse-mode differentiation [1] to quantum.

Fortunately, a series of recent research on analytical formulas of “quantum” gradients [9, 24, 31, 49, 59] has helped (partially) overcome these difficulties and thus enabled AD on quantum circuits, which has already been adopted in major quantum machine learning platforms, including Tensorflow Quantum [16] and PennyLane [12].

Zhu et al. [77] provide the first rigorous formalization of the AD technique for quantum programs with bounded loops beyond quantum circuits. They also leveraged their framework in the training of a VQC instance with controls, which has superior performance than normal VQCs for certain machine learning tasks.

Quantum Applications with Unbounded Loops. Most existing AD results in quantum computing have been focusing on applications of variational quantum circuits (or their variants) for a few designated tasks, which misses the opportunity to investigate more sophisticated quantum algorithms. For example, parameterized quantum programs with *unbounded loops* can describe a rich family of quantum algorithms with a few unspecified parameters, which could be trained to help quantum programs meet the runtime requirement, e.g., achieving quantum speedup in the examples of quantum walk [3] and amplitude amplification [15] or generating desired unitaries that are unknown beforehand in the example of the repeat-until-success unitary implementation [14]. Analytical derivation of these parameters, if ever possible, would likely require domain knowledge of the underlying problem and is usually done in a case-by-case fashion. Instance-driven gradient-based search of these parameters is a promising alternative, which is only possible with the AD technique for unbounded loops.

Let us dive into a simple example based on **amplitude amplification (AA)**. A direct adoption of the textbook AA would be written as a for-loop with a given number of iterations. However, this number of iterations is often hard to determine beforehand, which makes it desirable to write the algorithm as a while-loop (i.e., an unbounded loop) and let the program decide when to terminate.

To that end, a framework with while-loops and parameterized weak measurements has been introduced [4, 50] as the *parameterized AA* program in Figure 1, where parameter θ controls the coupling strength between the search variable q and the measure variable r . The choice of θ is critical in achieving quantum speedups. While an analytical solution of θ exists for certain quantum speedup [4], its optimal choice that minimizes the expected runtime is still unknown.

As shown in our case study, gradient-based methods in differentiable quantum programming could automatically identify a better choice of θ than existing literature [4, 50] without domain knowledge about the AA algorithm, the promise of which also extends to parameterized quantum random walks and repeat-until-success unitary implementation. This provides a strong motivation to develop the AD technique for *unbounded* quantum loops. However, there is no general AD solution for unbounded loops even in classical (imperative) programs [55], which questions the feasibility of our goal.

Indeed, as we elaborate on in Section 3, unbounded loops introduce serious challenges in AD for classical, probabilistic, and quantum programs. Moreover, unique features of quantum programs, such as the no-cloning theorem and the branching induced by measurements, further restrict the available AD techniques for quantum programs with unbounded loops.

Contributions. We overcome these challenges and develop a differentiable quantum programming framework for unbounded loops, with the following contributions:

- A formulation of parameterized quantum **while**-language with unbounded loops and a new parameterized unitary operation called the density operator exponentiation $e^{-it\sigma}$ of any density operator σ that allows the inclusion of more unitary gates. (Section 4)
- A sufficient condition (i.e., finite-dimensional program state space) for the differentiability of quantum programs with unbounded loops (Theorem 4.5). We also exhibit an example of non-differentiable infinite-dimensional quantum programs (Example 4.6) to demonstrate the difference between finite and infinite dimensional quantum programs for differentiation. (Section 4)
- An AD scheme for quantum programs with unbounded loops with two components: (1) **Differentiation on a Single-Occurrence of Parameter (DSOP)** for quantum circuits with respect to a parameter with a single occurrence; and (2) **Extension to Unbounded Loops (EUL)** for unbounded loops based on any DSOP. We contribute a new DSOP technique, called the *commutator-form rule* inspired by Reference [44] for general $e^{-i\theta H}$, with a more general applicability.¹ We also develop the code transformation and establish its correctness (Theorem 5.4). (Section 5)
- Implementation of our AD scheme with Python and Q# and discussion of its relevant sample efficiency, in which we provide an upper bound that matches the one of Zhu et al. [77] when there is no unbounded loop. (Section 6)
- Extensive case study on the gradient-based approach for automatically identifying unknown parameters in quantum algorithm design, which includes the parameterized AA algorithm,

```

while  $M[r] = 1$  do
   $q := \text{GroverRotation}[q]$ ;
   $q, r := \text{Coupling}_\theta[q, r]$ 
od

```

Fig. 1. Parameterized AA.

¹It removes the limitation of the parameter-shift rule that is only applicable when H has at most two distinct eigenvalues.

Table 1. A Summary of Notation Used in This Article

Sets, Spaces		States		Operators, Operations	
$\mathcal{H}, \mathcal{H}_q$	Hilbert space	$ \psi\rangle, 0\rangle, 1\rangle, +\rangle, -\rangle$	pure state	$U, V, e^{-i\theta\sigma},$ $H, X, Y, Z, X \otimes X$	unitary
$\mathcal{L}(\mathcal{H})$	linear operators of $\mathcal{H}$	$\rho, \sigma, \psi\rangle\langle\psi $	density operator	$M, \{M_m\}, \{ 0\rangle\langle 0 , 1\rangle\langle 1 \}$ $O = \sum_m \lambda_m \psi_m\rangle\langle\psi_m $	measurement observable
$\mathcal{D}(\mathcal{H})$	partial density operators of $\mathcal{H}$	ρ with $\text{tr}(\rho) \leq 1$	partial density operator	$\mathcal{E} : \rho \mapsto \sum_j E_j \rho E_j^\dagger$	superoperator

quantum walk with parameterized shift operator on 2D grids, and unitary implementation with parameterized repeat-until-success algorithms. (Section 7)

Related Work. There is a rich literature on differentiation rules of quantum circuits [5, 35, 39, 42, 49, 59, 65, 68]. These researches focus on how to use quantum hardware to derive the derivative of the expectation function of a parametrized quantum circuit. For Pauli rotations $U(\theta) = e^{-i\theta\Delta/2}$, $\Delta = X, Y, Z$, Li et al. [42] and Mitarai et al. [49] first proposed a formula that only needs to run the initial circuit twice with different parameters to find the derivative. Then, Schuld et al. [59] named this formula the “parameter-shift rule” and expanded it to a general case of $U(\theta) = e^{-i\theta H}$ with Hamiltonian H having at most two distinct eigenvalues. Recently, independent developments of variants of the parameter-shift rules (general parameter-shift rules) [35, 39, 68] were proposed for general Hamiltonian H . Their works can be traced back to an observation that the expectation function of a PQC with a single parameter is a finite Fourier series [65]. Our commutator-form rule, which is applicable to $e^{-i\theta H}$ for general H , is based on a very different technique and has a simple form compared to general parameter-shift rules.

Most existing AD techniques in quantum computing [12, 16, 21, 38, 46, 51] work with simple languages describing quantum circuits without any control flow. Some of these results, e.g., Yao. jl [46], also apply classical AD techniques to classical programs that simulate quantum circuits, which is, unfortunately, not scalable for real quantum applications.

The only exception and also the closest work to ours is Reference [77], which proposed differentiable quantum programming with bounded loops beyond quantum circuits. Although the syntax in Reference [77] supports general parameterized gates, its code transformation only supports Pauli rotation gates based on a variant of the parameter-shift rule. To handle AD of bounded quantum loops, Zhu et al. [77] used a finite collection of quantum programs and added up their outputs for the derivative. As elaborated on in Section 3, one cannot extend this approach to a collection of unbounded sizes like unbounded loops in this article. The correctness and feasibility of this article to deal with infinite summation caused by unbounded loops is the main difficulty, which Zhu et al. [77] did not encounter. Moreover, the efficiency of this article is comparable to Reference [77] in the bounded-loop setting. Thus, this article strictly improves Reference [77].

2 QUANTUM PRELIMINARIES

In this section, we recall some basic knowledge of quantum computing and provide a summary of notation in Table 1. The reader can consult the standard textbook [52, Chapter 2, 4] for more details.

2.1 States and Hilbert Spaces

The state space of an isolated quantum system is represented by a complex Hilbert space. We use the Dirac notation $|\psi\rangle$ to denote a (column) vector in a Hilbert space. The (vector dual) Hermitian conjugate of $|\psi\rangle$ is (a row vector) denoted by $\langle\psi|$. The inner product of $|\psi\rangle$ and $|\phi\rangle$ is denoted by

$\langle\phi|\psi\rangle$, considered as a shorthand for $\langle\phi|(|\psi\rangle)$. The norm of a vector $|\psi\rangle$ is defined as $\| |\psi\rangle \| = \sqrt{\langle\psi|\psi\rangle}$. A unit vector is referred to as a *pure state*.

Example 2.1 (Qubit System). The state space of a quantum bit (qubit) is a 2-dimensional Hilbert space $\mathcal{H}_2 = \mathbb{C}^2$ with $|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ and $|1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ being the *computational basis*. A pure state $|\psi\rangle \in \mathcal{H}_2$ can be expressed as

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \text{ with } |\alpha|^2 + |\beta|^2 = 1.$$

There are also two states of the qubit system that often appear: $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$, $|-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$. They also form a basis of $\mathcal{H}_2$.

A (linear) operator is a linear mapping between Hilbert spaces, and the set of all operators from $\mathcal{H}$ to $\mathcal{H}'$ is denoted by $\mathcal{L}(\mathcal{H}, \mathcal{H}')$. Specifically, an operator $A \in \mathcal{L}(\mathcal{H}, \mathcal{H})$ is said to be an operator on $\mathcal{H}$ and write $A \in \mathcal{L}(\mathcal{H})$. We often write $I_{\mathcal{H}}$ for the identity operator on $\mathcal{H}$.

The Hermitian conjugate (adjoint) of an operator A is denoted by $A^\dagger$. An operator A on $\mathcal{H}$ is *Hermitian* if $A^\dagger = A$. An operator A on $\mathcal{H}$ is *positive semidefinite* if for all vectors $|\psi\rangle \in \mathcal{H}$, $\langle\psi|A|\psi\rangle \geq 0$. The Löwner order $\sqsubseteq$ is defined as $A \sqsubseteq B$ if $B - A$ is positive semidefinite. The *trace* of an operator A on $\mathcal{H}$ is defined as $\text{tr}(A) = \sum_j \langle\psi_j|A|\psi_j\rangle$, with $\{|\psi_j\rangle\}$ an orthonormal basis of $\mathcal{H}$.

Example 2.2 (Outer Product). The outer product of two states $|\psi\rangle, |\phi\rangle \in \mathcal{H}$, denoted by $|\psi\rangle\langle\phi|$, is an operator on $\mathcal{H}$ defined as $(|\psi\rangle\langle\phi|)(|\varphi\rangle) = |\psi\rangle \cdot (\langle\phi|\varphi\rangle) = \langle\phi|\varphi\rangle |\psi\rangle$ for any $|\varphi\rangle \in \mathcal{H}$. In particular, the trace of $|\psi\rangle\langle\phi|$ is

$$\sum_j \langle\psi_j|(|\psi\rangle\langle\phi|)|\psi_j\rangle = \sum_j \langle\psi_j|\psi\rangle\langle\phi|\psi_j\rangle = \sum_j \langle\phi|\psi_j\rangle\langle\psi_j|\psi\rangle = \langle\phi|\left(\sum_j |\psi_j\rangle\langle\psi_j|\right)|\psi\rangle = \langle\phi|I_{\mathcal{H}}|\psi\rangle = \langle\phi|\psi\rangle.$$

For example, the operator $|0\rangle\langle-|$ maps $|1\rangle$ to $\langle-|1\rangle|0\rangle = -\frac{1}{\sqrt{2}}|0\rangle$ and $\text{tr}(|0\rangle\langle-|) = \langle-|0\rangle = \frac{1}{\sqrt{2}}$, which can be illustrated in matrix multiplication as

$$\begin{aligned} |0\rangle\langle-|(1) &= \begin{pmatrix} 1 \\ 0 \end{pmatrix} \cdot \begin{pmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 1 \end{pmatrix} = -\frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = -\frac{1}{\sqrt{2}}|0\rangle, \\ \text{tr}(|0\rangle\langle-|) &= \text{tr}\left(\begin{pmatrix} 1 \\ 0 \end{pmatrix} \cdot \begin{pmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix}\right) = \frac{1}{\sqrt{2}}. \end{aligned}$$

When the state of a quantum system is not completely known, people may think of it as a *mixed state* (ensemble of pure state) $\{(p_j, |\psi_j\rangle)\}$ meaning that it is at $|\psi_j\rangle$ with probability p_j . A *density operator* for this system is defined as $\rho = \sum_j p_j |\psi_j\rangle\langle\psi_j|$. Formally, a density operator ρ on a Hilbert space $\mathcal{H}$ is a positive semidefinite operator with $\text{tr}(\rho) = 1$. Moreover, a *partial density operator* ρ on $\mathcal{H}$ is defined as a positive semidefinite operator with $\text{tr}(\rho) \leq 1$. We use $\mathcal{D}(\mathcal{H})$ to denote the set of partial density operators on $\mathcal{H}$.

2.2 Quantum Operations

Unitary Transformations. An operator U on a Hilbert space $\mathcal{H}$ is a *unitary transformation* if $U^\dagger U = U U^\dagger = I_{\mathcal{H}}$. A unitary transformation U describes the evolution from any pure state $|\psi\rangle$ to $U|\psi\rangle$. For mixed states, this evolution is reformulated as from any mixed state ρ to $U\rho U^\dagger$.

Example 2.3 (Common Single-qubit Unitaries). Common single-qubit unitary operators include H (Hadamard gate) and X, Y, Z (Pauli gates). Their matrix representation with respect to basis $\{|0\rangle, |1\rangle\}$ are:

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

The H gate can transform the computational basis $\{|0\rangle, |1\rangle\}$ and the basis $\{|+\rangle, |-\rangle\}$ into each other as $H|0\rangle = |+\rangle, H|1\rangle = |-\rangle$ and $H|+\rangle = |0\rangle, H|-\rangle = |1\rangle$. In addition, we can write H in the form of outer products as $H = |+\rangle\langle 0| + |-\rangle\langle 1| = |0\rangle\langle +| + |1\rangle\langle -|$. The X gate acts as a “Not” gate, exchanging $|0\rangle$ and $|1\rangle$, i.e., $X|0\rangle = |1\rangle, X|1\rangle = |0\rangle$; thus, we can write it in the form of outer products as $X = |1\rangle\langle 0| + |0\rangle\langle 1|$. Similarly, we can also write $Y = i|1\rangle\langle 0| - i|0\rangle\langle 1|, Z = |0\rangle\langle 0| - |1\rangle\langle 1|$.

Measurements and Observables. A measurement on a system with a state space $\mathcal{H}$ is described by a collection $\{M_m\}$ of *measurement operators* on $\mathcal{H}$ with the *completeness equation*: $\sum_m M_m^\dagger M_m = I_{\mathcal{H}}$. When performing a measurement $\{M_m\}$ on a pure state $|\psi\rangle$ and a mixed state ρ , the outcome of index m occurs with probability $p(m) = \langle\psi|M_m^\dagger M_m|\psi\rangle$ and $p(m) = \text{tr}(M_m \rho M_m^\dagger)$, the corresponding state of the system after the measurement is $|\psi_m\rangle = M_m|\psi\rangle/\sqrt{p(m)}$ and $\rho_m = M_m \rho M_m^\dagger/p(m)$, respectively. In the context of mixed states, if we do not know the outcome of the measurement, the state of the system after the measurement can be described by $\sum_m p(m)\rho_m = \sum_m M_m \rho M_m^\dagger$.

A projective measurement is often described by an *observable*, M , a Hermitian operator on $\mathcal{H}$. The spectral decomposition of $M = \sum_m m P_m$ corresponds to a quantum measurement $\{P_m\}$ with measurement outcome m for each P_m . The average value of this measurement performed on a state $|\psi\rangle$ is $\langle\psi|M|\psi\rangle$. The value $\langle\psi|M|\psi\rangle$ is often written as $\langle M \rangle$ and called the *expectation* of M . For a mixed state ρ , the expectation of M is $\text{tr}(M\rho)$.

Example 2.4 (Pauli Measurements). The Pauli gates X, Y, Z are also Hermitian and therefore are observables. For example, X admits a spectral decomposition $X = |+\rangle\langle +| - |-\rangle\langle -|$, then it describes the measurement $M = \{M_1 = |+\rangle\langle +|, M_{-1} = |-\rangle\langle -|\}$. For a mixed state $\rho = \frac{1}{4}|0\rangle\langle 0| + \frac{3}{4}|1\rangle\langle 1|$, the measurement M will result in the state $|+\rangle\langle +|$ with probability $\text{tr}(|+\rangle\langle +|\rho) = \frac{1}{2}$ and the state $|-\rangle\langle -|$ with probability $\frac{1}{2}$. Moreover, the measurement outcome will be 1 with probability $\frac{1}{2}$ and -1 with probability $\frac{1}{2}$, thus the expectation of the measurement outcome is $0 = \frac{1}{2} - \frac{1}{2} = \text{tr}(|+\rangle\langle +|\rho) - \text{tr}(|-\rangle\langle -|\rho) = \text{tr}(X\rho)$.

General Quantum Operations. A *superoperator* is a linear mapping between $\mathcal{L}(\mathcal{H})$ and $\mathcal{L}(\mathcal{H}')$. For mixed states, unitary transformations and measurements can be described by a general form of completely positive and trace-non-increasing superoperators, which has the *Kraus representation*: $\sum_j E_j(\cdot)E_j^\dagger$ with $E_j \in \mathcal{L}(\mathcal{H}, \mathcal{H}')$ and $\sum_j E_j^\dagger E_j \subseteq I_{\mathcal{H}}$ [69]. The Schrödinger-Heisenberg dual of a superoperator $\mathcal{E}$ with the Kraus representation $\mathcal{E}(\cdot) = \sum_j E_j(\cdot)E_j^\dagger$ is $\mathcal{E}^*(\cdot) = \sum_j E_j^\dagger(\cdot)E_j$.

2.3 Composite Systems and Tensor Products

The tensor product of two vectors $|\psi_1\rangle$ and $|\psi_2\rangle$ is denoted by $|\psi_1\rangle \otimes |\psi_2\rangle$, which is sometimes written as $|\psi_1\rangle|\psi_2\rangle$ or even $|\psi_1\psi_2\rangle$ for short. The tensor product of two Hilbert spaces $\mathcal{H}_1$ and $\mathcal{H}_2$ is denoted by $\mathcal{H}_1 \otimes \mathcal{H}_2$. For any linear operator $A_1 \in \mathcal{L}(\mathcal{H}_1)$ and $A_2 \in \mathcal{L}(\mathcal{H}_2)$, their tensor product operator $A_1 \otimes A_2 \in \mathcal{L}(\mathcal{H}_1 \otimes \mathcal{H}_2)$ is defined by linear extensions of $A_1 \otimes A_2(|\psi_1\rangle \otimes |\psi_2\rangle) = (A_1|\psi_1\rangle) \otimes (A_2|\psi_2\rangle)$ for any $|\psi_1\rangle \in \mathcal{H}_1, |\psi_2\rangle \in \mathcal{H}_2$.

The state space of a composite quantum system is the tensor product of its components' state spaces, e.g., if a system with two components in state $|\psi_1\rangle \in \mathcal{H}_1$ and state $|\psi_2\rangle \in \mathcal{H}_2$, respectively, then the joint state of the composite system is $|\psi_1\rangle \otimes |\psi_2\rangle \in \mathcal{H}_1 \otimes \mathcal{H}_2$. For mixed states, if a system with two components in the state $\rho_1 \in \mathcal{D}(\mathcal{H}_1)$ and the state $\rho_2 \in \mathcal{D}(\mathcal{H}_2)$, respectively, then the joint state of the composite system is $\rho_1 \otimes \rho_2 \in \mathcal{D}(\mathcal{H}_1 \otimes \mathcal{H}_2)$.

For two Hilbert spaces $\mathcal{H}_1, \mathcal{H}_2$ and any operator $A \in \mathcal{L}(\mathcal{H}_1 \otimes \mathcal{H}_2)$, the *partial trace* over space $\mathcal{H}_2$ of A is $\text{tr}_{\mathcal{H}_2}(A) = \sum_j (I_{\mathcal{H}_1} \otimes \langle\psi_j|)A(I_{\mathcal{H}_1} \otimes |\psi_j\rangle) \in \mathcal{L}(\mathcal{H}_1)$, where $\{|\psi_j\rangle\}$ is an orthonormal basis of $\mathcal{H}_2$ and we often write tr_2 for $\text{tr}_{\mathcal{H}_2}$ if there is no ambiguity. The notion of partial trace can be used to describe sub-systems of a composite quantum system. Suppose we have a composite

$r := k;$	$r := 0 \oplus_{0.5} 1;$	$r := (0\rangle - 1\rangle)/\sqrt{2};$	$r := (0\rangle - 1\rangle)/\sqrt{2};$
while $r \neq 0$ do	while $r \neq 0$ do	while $M[r] \neq 0$ do	while ^(k) $M[r] \neq 0$ do
$r := r - 1$	$r := 0 \oplus_{0.5} 1$	$r := H[r]$	$r := H[r]$
od	od	od	od
(a) Classical.	(b) Probabilistic.	(c) Quantum.	(d) Quantum, bounded.

Fig. 2. Four while-loop programs to demonstrate unbounded loops in quantum programs.

system with two components q_1 and q_2 , whose state spaces are $\mathcal{H}_1$ and $\mathcal{H}_2$, respectively, and the whole state of the composite system is $\rho \in \mathcal{D}(\mathcal{H}_1 \otimes \mathcal{H}_2)$, then the state of component system q_1 is a reduced density operator defined as $\text{tr}_2(\rho)$.

Example 2.5 (n-qubit System). The state space of n -qubit system is $\mathcal{H} = (\mathbb{C}^2)^{\otimes n} \cong \mathbb{C}^{2^n}$, which is the tensor product of n copies of the state space of single qubit, with $\{|x\rangle \mid x \in \{0, 1\}^n\}$ being the computational basis. Thus, an n -qubit pure state $|\psi\rangle$ can be expressed as $\sum_{x \in \{0, 1\}^n} \alpha_x |x\rangle$ with $\sum_{x \in \{0, 1\}^n} |\alpha_x|^2 = 1$. For example, an important 2-qubit state is the Bell state $|\beta_{00}\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \in \mathbb{C}^2 \otimes \mathbb{C}^2$. This state embodies *quantum entanglement*, because it cannot be written as $|\psi_1\rangle \otimes |\psi_2\rangle$ for any $|\psi_1\rangle, |\psi_2\rangle \in \mathbb{C}^2$.

3 CHALLENGES AND OUR KEY IDEAS

Let us revisit classical, probabilistic, and quantum programs as shown in Figure 2 with one unbounded loop to understand their features, differences, and corresponding difficulties in differentiation. We assume some simple quantum terminology and refer readers to a more detailed preliminary in Section 4.

While-loop, an important construct making an imperative programming language Turing-complete, may cause an arbitrary number of loops or infinite loops (not terminated). In *classical (deterministic) programs*, such as Figure 2(a), the variable r is assigned with an integer k . When $k \geq 0$, the while-loop will execute the loop body k times and terminate, and when $k < 0$, the while-loop will execute the loop body for infinitely many times and not terminate. Nevertheless, due to the deterministic nature of classical programs, for fixed inputs, *there is one and only one path of the program execution*.

However, there may be an infinite number of execution paths in a *quantum program* or *probabilistic program*. In Figure 2(b), the command $r := 0 \oplus_{0.5} 1$ assigns 0 to variable r with probability 0.5 or 1 to r with probability 0.5 otherwise. Thus, the probability of the probabilistic program in Figure 2(b) executing the loop body k , $k \geq 0$, times is 0.5^{k+1} , which means this program has an *infinite* number of execution paths. Similarly, the quantum program in Figure 2(c) also has an *infinite* number of execution paths. Let us see the execution of the program in Figure 2(c):

- (1) First, r is assigned with state $|-\rangle = (|0\rangle - |1\rangle)/\sqrt{2}$.
- (2) Second, in the measurement of the while-loop, state $|-\rangle$ is measured with $\{M_0 = |0\rangle\langle 0|, M_1 = |1\rangle\langle 1|\}$. The measurement outcome will be 0 *with probability* $\langle -|M_0|-\rangle = 0.5$ and 1 *with probability* $\langle -|M_1|-\rangle = 0.5$. When the outcome is 0, the program will terminate; when the outcome is 1, the measured state of r will become $|1\rangle$, and the program will enter the loop body (3).
- (3) In the loop body, applied with H , the state of r becomes $|-\rangle = H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$, then the program goes back to the measurement of the while-loop (2).

Table 2. Comparisons among Classical, Probabilistic, and Quantum Programs

	Classical	Probabilistic	Quantum
# of Execution Paths	one	possibly infinitely many	possibly infinitely many
Distribution over Paths	none	explicit by sampling	implicit by measurement
Usability of Chain Rule	yes	yes	no
Differentiability	almost everywhere [48]	almost everywhere for every path [47], sufficient conditions for bounded loops [40]	sufficient condition for unbounded loops , Theorem 4.5

From (2) above, we can see that the program will terminate with probability 0.5 or continue the while-loop with probability 0.5 at each entry of the while-loop. Therefore, the probability of this program executing loop body ($r := H[r]$) for $k, k \geq 0$, times is 0.5^{k+1} , which means this program has an *infinite* number of execution paths. Note that the probability 0.5 is implicitly implied by the measurement outcome of the quantum program, as the quantum no-cloning theorem prevents us from accurately tracking the intermediate states of the quantum programs. However, in probabilistic programs, such probabilities are explicitly given by the sampling primitives. (Even if the sampled distribution depends on some parameters, e.g., Gaussian distribution $\mathcal{N}(\mu, \sigma)$ with μ, σ generated at runtime, we can still know the specific distribution of the sample by recording the values of these parameters at runtime.)

Another important difference worth highlighting between classical (deterministic or probabilistic) programs and quantum ones is the quantum no-cloning theorem that prohibits the use of the *Chain-rule-based* forward/reverse differentiation in the quantum setting. For example, consider a simple classical program $u := g(x); y := f(u)$. The final value of y is $f(g(x))$. According to the Chain rule, the derivative of y with respect to x is $\frac{dy}{dx} = \frac{dy}{du} \cdot \frac{du}{dx} = \frac{df}{du}(g(x)) \cdot \frac{dg}{dx}(x)$. The classical forward-AD with dual number (e.g., Reference [7]) will introduce a intermediate variable $\hat{v}$ (the intermediate derivative of this variable with respect to x) for each variable v and lift each function h to $\hat{h} : (v, \hat{v}) \mapsto (h(v), \frac{dh}{dv}(v) \cdot \hat{v})$, which follows the Chain rule. Then, we obtain a new program $(u, \hat{u}) := \hat{g}(x, 1); (y, \hat{y}) := \hat{f}(u, \hat{u})$, from which we can compute that $\hat{y} = \frac{df}{du}(u)\hat{u} = \frac{df}{du}(g(x))\frac{dg}{dx}(x)$. Thus, this new program achieves AD. For each lifted function $\hat{h}$, the input v is fed not only h but also $\frac{dh}{dv}$, in which an implicit copy of v is made. However, the quantum no-cloning theorem only allows v to be fed into h or $\frac{dh}{dv}$ if we think of them as “quantum.” As a result, AD techniques in quantum are somewhat separate from those commonly studied in the classical AD literature.

Finally, consider a *bounded-loop quantum program* in Figure 2(d), which is investigated in Reference [77]. The number $k > 0$ in **while**^(k) limits the iteration times of the loop body up to k , which results that the program in Figure 2(d) has at most $k + 1$ and hence a *finite* number of execution paths.

We summarize these comparisons in Table 2. As we will see, dealing with infinitely many execution paths is one major difficulty in differentiation over unbounded loops, either probabilistic or quantum. The unusability of the Chain rule further complicates the quantum case.

3.1 Differentiability of Unbounded Quantum Loops

The *differentiability* of unbounded quantum loops should be the first question to address, as it is already quite non-trivial in establishing so in classical and probabilistic functional programs.

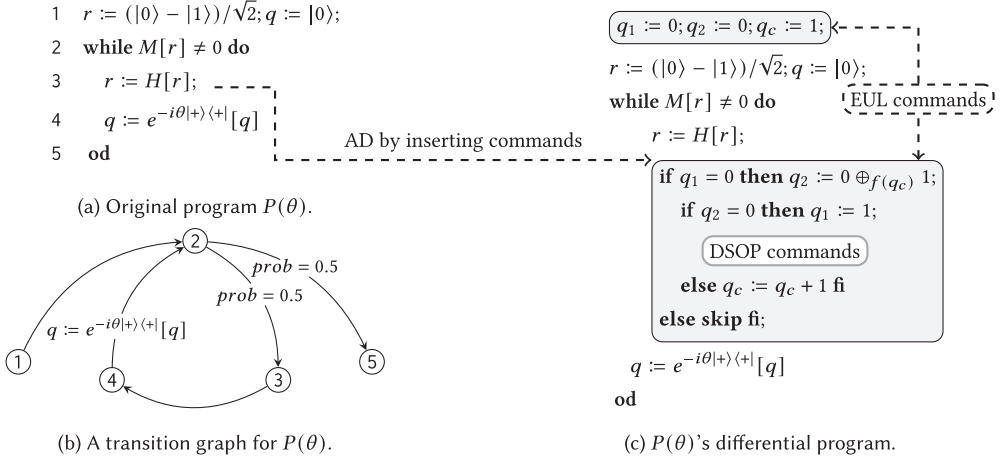


Fig. 3. Running example to demonstrate our AD scheme. Every time the program runs to a statement, e.g., $q := e^{-i\theta|+\rangle\langle +|}[q]$ here, that contains θ , it will first enter a block of EUL to decide whether to continue $P(\theta)$ or to do a differentiation operation by DSOP then continue $P(\theta)$. $0 \oplus_p 1$ is a probabilistic choice that outputs 0 with probability p and 1 with probability $1 - p$ for any $0 \leq p \leq 1$, and $f(n) = \mu(n)/(1 - \sum_{j=1}^{n-1} \mu(j))$ is determined by the distribution μ mentioned in Section 3.2.

In classical programs, conditionals often lead to piecewise-defined functions and non-differentiable points, which are discontinuous or have different left and right derivatives [8]. Even if the function defined by a program is differentiable, syntactic discontinuity can make AD fail.² To resolve the issue of conditionals, Abadi and Plotkin [1] adopted the “partial conditionals,” which ignores the boundary case, and proved the correctness of AD on conditionals and recursion. Mazza and Pagani [48] characterized the set of “stable points,” the intuition behind which is the point that has an open neighborhood with the same execution trace (“execution path”). They proved that AD is almost everywhere correct under the mild hypothesis. However, these arguments are developed for one execution path in classical programs.

The case of probabilistic programs resembles quantum programs a lot due to possibly infinitely many execution paths. Whether probabilistic programs would lead to non-differentiable densities at some non-measure-zero set has been an important open question in the field [72]. Recently, Mak et al. [47] considered higher-order probabilistic programming with recursion and proved that a probabilistic program’s density is almost everywhere differentiable under mild hypothesis. However, it only tells us the differentiability of any trace of sampled values during execution, which implies a fixed execution path.

Recall the program in Figure 2(c), on which we add a command with parameter θ into the loop body as our running example in Figure 3. The transition graph for each line of program $P(\theta)$ in Figure 3(a) is shown in Figure 3(b), where the behavior of $P(\theta)$ is the same as in Figure 2(c): $P(\theta)$ will terminate (goto line 5) with probability 0.5 or continue the while-loop (goto line 3) with probability 0.5 at each entry (line 2) of the while-loop. The probability of $P(\theta)$ executing k times

²Consider the program presented in References [1, 48]:

$$\text{SillyId} \equiv \lambda x^{\mathbb{R}}. \text{if } x = 0 \text{ then } 0 \text{ else } x.$$

We can see that $\llbracket \text{SillyId} \rrbracket(x) = x$, thus $\llbracket \text{SillyId} \rrbracket$ has a constant derivative of 1. However, general AD would produce the wrong answer 0 at the point $x = 0$.

$q := e^{-i\theta|+\rangle\langle +|}[q]$ is 0.5^{k+1} . Consider operations related to variable q , $P(\theta)$ induces a semantic function like

$$f_k(\theta) = \llbracket q := e^{-i\theta|+\rangle\langle +|}[q] \rrbracket^k = \underbrace{\llbracket q := e^{-i\theta|+\rangle\langle +|}[q] \rrbracket \circ \dots \circ \llbracket q := e^{-i\theta|+\rangle\langle +|}[q] \rrbracket}_{k-1 \text{ times function composite}}$$

with probability 0.5^{k+1} , $k \geq 0$. The differentiability of $f_k(\theta)$ is generally easily obtainable, while the differentiability of its expectation $F(\theta) = \sum_{k=0}^{\infty} f_k(\theta)/2^{k+1}$ is unclear. For this problem of expectation, two conditions are proposed in the probabilistic programming [40]:

— *Differentiability of expectation (infinite summation):*

$$F(\theta) = \sum_k v(\theta, k) f_k(\theta) \text{ is differentiable on } \mathbb{R}. \quad (\text{Condition A1})$$

— *Exchangeability between differentiation and infinite summation:*

$$\text{for all } \theta \in \mathbb{R}, \sum_k v(\theta, k) f_k(\theta) = \sum_k \partial_\theta (v(\theta, k) f_k(\theta)), \quad (\text{Condition A2})$$

where $v(\theta, \cdot)$ is a probability distribution over index k . [Condition A1](#) states the premise of AD: We would not talk about AD if the function induced by the program was not differentiable. [Condition A2](#) states that the differential operation on every trace ($\partial_\theta (v(\theta, k) f_k(\theta))$, which reflects the underlying AD implementation) can be collected into the differential operation on the total program ($\partial_\theta \sum_k v(\theta, k) f_k(\theta)$). To the best of our knowledge, no research has yet investigated what kind of probabilistic programs or quantum programs with unbounded loops meet [Condition A1](#) and [Condition A2](#).

Our Solution. Fortunately, we identify *finite-dimensional state space*, which is met by all existing quantum applications, as a sufficient condition to satisfy [Condition A1](#) and [Condition A2](#). Technically, under this condition, the probability that an unbounded quantum loop iterates k times has an exponential decay on k (Lemma 5.3). The cornerstone behind this lemma is the *compactness* of finite-dimensional Hilbert spaces. As in our running example, $f_k(\theta) = \llbracket q := e^{-i\theta|+\rangle\langle +|}[q] \rrbracket^k$, which corresponds to loop k times, appears with probability 0.5^{k+1} . Let g_θ denote $\llbracket q := e^{-i\theta|+\rangle\langle +|}[q] \rrbracket$, the differential of $f_k(\theta)$ is a summation of differentiation at every occurrence of θ , i.e., $\partial_\theta f_k(\theta) = \sum_{j=1}^k (g_\theta)^{j-1} \circ (\partial_\theta g_\theta) \circ (g_\theta)^{k-j}$, then

$$\sum_{k=0}^{\infty} \partial_\theta \left(\frac{f_k(\theta)}{2^{k+1}} \right) = \sum_{k=0}^{\infty} \frac{1}{2^{k+1}} \sum_{j=1}^k (g_\theta)^{j-1} \circ (\partial_\theta g_\theta) \circ (g_\theta)^{k-j} \quad (3.1)$$

is uniformly convergent as $\frac{1}{2^{k+1}} \sum_{j=1}^k (g_\theta)^{j-1} \circ (\partial_\theta g_\theta) \circ (g_\theta)^{k-j} \in O(\frac{k}{2^{k+1}})$. This uniform convergence implies both [Condition A1](#) and [Condition A2](#).

Conversely, through Weierstrass' non-differentiable function $S(x) = \sum_{n=0}^{\infty} a^n \sin(b^n x)$ [33, Theorem 1.31], we can construct a counterexample (Example 4.6) that is nowhere differentiable on $\mathbb{R}$ for quantum loops with infinite-dimensional space. Specifically, Example 4.6 has two nested loops that induces $f_k(\theta) = \llbracket q := e^{-i\theta|+\rangle\langle +|}[q] \rrbracket^{2^k}$ with probability 0.5^{k+1} . The differential of $f_k(\theta)$ becomes $\partial_\theta f_k(\theta) = \sum_{j=1}^{2^k} (g_\theta)^{j-1} \circ (\partial_\theta g_\theta) \circ (g_\theta)^{2^k-j}$, then this summation of 2^k terms makes $\sum_{k=0}^{\infty} \frac{1}{2^{k+1}} \partial_\theta f_k(\theta) = \sum_{k=0}^{\infty} \frac{1}{2^{k+1}} \sum_{j=1}^{2^k} (g_\theta)^{j-1} \circ (\partial_\theta g_\theta) \circ (g_\theta)^{2^k-j}$ divergent everywhere. Note that the probability 0.5^{k+1} comes from the outer loop, and the exponent 2^k comes from the inner loop. For the latter to happen, the inner loop, like Example 4.6, needs to have an infinite-dimensional register to record the information of k , which goes to positive infinite. This non-differentiable

counterexample can also be represented by the expectation of a probabilistic program (see Example 4.6).

3.2 Execution Paths with Infinitely Many Parameter Occurrences

As we saw in Figure 3, loops lead to repeated execution of loop body, which means a parameter can appear many times at a single execution path, e.g., the function $f_k(\theta) = \llbracket q := e^{-i\theta|+}\langle + | [q] \rrbracket^k$ introduced in Section 3.1 has k occurrences of θ . These execution paths refer to *quantum circuits with a parameter appearing multiple times*. In the case of bounded quantum loops, the total number of execution paths is finite, and the multiplicity of parameter occurrences on each path is also bounded. Therefore, Zhu et al. [77] proposed additive quantum programs to represent a (finite) collection of quantum programs that compute partial derivatives of all occurrences. Conceivably, this approach does not extend to the infinite (and unbounded) case.

Our Solution. According to Section 3.1, the differential of $F(\theta) = \sum_{k=0}^{\infty} f_k(\theta)/2^{k+1}$ should be equal to

$$\sum_{k=0}^{\infty} \frac{\partial_{\theta} f_k(\theta)}{2^{k+1}} = \sum_{k=0}^{\infty} \sum_{j=1}^k \frac{1}{2^{k+1}} (g_{\theta})^{j-1} \circ (\partial_{\theta} g_{\theta}) \circ (g_{\theta})^{k-j} = \sum_{j=1}^{\infty} \sum_{k=1}^{\infty} \frac{1}{2^{k+1}} (g_{\theta})^{j-1} \circ (\partial_{\theta} g_{\theta}) \circ (g_{\theta})^{k-j}. \quad (3.2)$$

Here, the second equality, i.e., the commutativity of the summation indexes j, k , is guaranteed by uniform convergence, as we mentioned for Equation (3.1). We rewrite Equation (3.2), as $\sum_{j=1}^{\infty} a_j$ with $a_j = \sum_{k=1}^{\infty} \frac{1}{2^{k+1}} (g_{\theta})^{j-1} \circ (\partial_{\theta} g_{\theta}) \circ (g_{\theta})^{k-j}$ is the corresponding summation term for all execution paths with differentiation on j th position. With the infeasibility of the Chain rule, the existing quantum AD technique can only handle a_j one-by-one, the cost of which will depend on the number of terms in Equation (3.2) that will be unbounded in our case.

Inspired by the *importance sampling* in statistics [61], we construct a random variable X such that

$$\Pr(X = a_j/\mu(j)) = \mu(j), \forall j \in \mathbb{Z}_+$$

to estimate the infinite sum $\sum_j^{\infty} a_j$, where μ is a probability distribution on $\mathbb{Z}_+$. By construction, the expectation of X is $\sum_{j=1}^{\infty} a_j$. Let us now focus on Figure 3(c), where the role of EUL commands, written with probabilistic pseudocode, is to generate the distribution μ as the original program runs and to attribute this probabilistic distribution to DSOP commands (differentiation operations that induce a_j) in the execution order. Precisely, the variable q_c records the number of loops until $q_2 = 0$. Together with the probabilistic choice $q_2 := 0 \oplus_{f(q_c)} 1$ (see definition in Figure 3's caption), the probability of $q_c = j$, $q_2 = 0$ and $q_1 = 1$, which means DSOP commands (a differentiation operation) are executed, and one term of a_j is evaluated, is $\mu(j)$ when fixing an execution path of $P(\theta)$ with loops' number $n > j$. Therefore, we associate the probability $\mu(j)$ with a_j in the execution order. Then, the desired random variable X is natural to construct.

However, for the estimation efficiency of X 's expectation, the variance of X should also be bounded. To that end, we identify a sufficient condition for the distribution $\mu : \mathbb{Z}_+ \rightarrow [0, 1]$ as

$$\lim_{n \rightarrow \infty} \sqrt[n]{\mu(n)} = 1, \quad (\text{converging-rate condition})$$

which would imply the correctness of our code-transformation (Theorem 5.4) and its efficiency (Theorem 6.2). Another implicit but critical property of our construction of random variable X in Figure 3(c) is its independence of the underlying execution path. It allows us to apply a simple and uniform code transformation while keeping all existing quantum branches that lead to all execution paths.

Applicability to classical and probabilistic programs with unbounded loops?

Ignoring the issue of differentiability, our idea of constructing random variables to sample partial derivatives can be applied to classical and probabilistic unbounded loops. However, it would be less efficient, because partial derivatives can be collected and forward/backward-propagated along the execution path by the Chain rule in the classical setting, which is not leveraged by our scheme.

Consider the example in Figure 4 where we show how EUL is applied to a probabilistic program. Denote the expectation of variable y after executing $P(\theta)$ in Figure 4(a) as $E(\theta) = \sum_{j=1}^{\infty} j \sin(\theta) \cdot 0.5^j = 2 \sin(\theta)$. Its derivative is $E'(\theta) = 2 \cos(\theta)$. Since this program $P(\theta)$ only uses a simple sampling primitive $r := 0 \oplus_{0.5} 1$,³ we can directly apply the classical forward-AD with a dual number (e.g., Reference [7]) as in Figure 4(b). Then, the expectation of $\hat{y}$ (dual number of y) after executing the program in Figure 4(b) is $\sum_{j=1}^{\infty} j \cos(\theta) \cdot 0.5^j = 2 \cos(\theta) = E'(\theta)$.

As a comparison, the rewriting of our EUL to the probabilistic program $P(\theta)$ becomes Figure 4(c). With the probabilistic choice $0 \oplus_{f(q_c)} 1$ and a variable q_c to count the occurrences of θ , the probability of the program executes command $(*)$ and the output of $q_c = j \geq 1$ is $\mu(j) \cdot 0.5^{j-1}$ if the output of q_c is $j \geq 1$. Then, the probability of $\hat{y} = \cos(\theta), q_c = j$ after executing the program in Figure 4(c) is $\mu(j) \cdot 0.5^{j-1}$. We can construct a random variable X with respect to $\hat{y}$ and q_c satisfies that $\Pr(X = \hat{y}/\mu(j)) \equiv \Pr(\hat{y} = \cos(\theta), q_c = j) = \mu(j) \cdot 0.5^{j-1}$. Hence, the expectation of X is $\sum_{j=1}^{\infty} \cos(\theta)/\mu(j) \cdot \mu(j) \cdot 0.5^{j-1} = \sum_{j=1}^{\infty} \cos(\theta) \cdot 0.5^{j-1} = 2 \cos(\theta)$, which is consistent with the above $E'(\theta)$. The major difference between Figures 4(b) and 4(c) is that Figure 4(b) can execute the command $(*)$ multiple times, while Figure 4(c) only executes $(*)$ once.

4 PARAMETERIZED QUANTUM WHILE-PROGRAMS

In this article, we expand a parameterized extension [77] of quantum **while**-language [73] to include unbounded loops.

4.1 Syntax

Let us first define the syntax of our programming language. Similar to References [73, 77], we assume a countably infinite set $\mathbf{qVar}$ of quantum variables and use the symbols $q, q', q_0, q_1, \dots \in \mathbf{qVar}$ as metavariables ranging over them. Each quantum variable $q \in \mathbf{qVar}$ has a type of Hilbert space $\mathcal{H}_q$ as its state space. A quantum register $\bar{q} = q_1, q_2, \dots, q_n$ is a finite sequence of distinct quantum variables, and its state space is $\mathcal{H}_{\bar{q}} = \bigotimes_{j=1}^n \mathcal{H}_{q_j}$.

Definition 4.1 (Syntax). A k -parameterized quantum **while**-program with parameter $\theta \in \mathbb{R}^k$ is generated by the syntax:

$$\begin{aligned} P(\theta) ::= & \text{skip} \mid q := |0\rangle \mid \bar{q} := U[\bar{q}] \mid \bar{q} := e^{-i\theta\sigma}[\bar{q}] \mid \\ & P_1(\theta); P_2(\theta) \mid \text{if } (\Box m \cdot M[\bar{q}] = m \rightarrow P_m(\theta)) \text{ fi} \mid \\ & \text{while } M[\bar{q}] = 1 \text{ do } P(\theta) \text{ od.} \end{aligned}$$

³For general probabilistic programs, sampling primitives that depend on the variable being differentiated, as well as the guards of conditionals that depend on the variable being differentiated, can make the probability distribution of the execution paths related to the variable. In such cases, we have to consider the “derivative” of the probability distribution of the execution paths. Recently, Lew et al. [41] resolve this problem for an expressive and higher-order probabilistic programming language (without general recursion) by equipping each sampling primitive with a built-in derivative estimation procedure. For quantum programs, the denotational semantics defined in Section 4.2 encode the probability of execution paths into density operators, thus, we do not need to treat the probability of execution paths separately.

```
r := 0; y := 0;
```

```
while r = 0 do
```

```
  y := y + sin( $\theta$ );
```

```
  r := 0  $\oplus_{0.5}$  1
```

```
od
```

(a) Original program $P(\theta)$.

```
r,  $\hat{r}$  := 0, 0; y,  $\hat{y}$  := 0, 0;
```

```
while r = 0 do
```

```
  y,  $\hat{y}$  := y + sin( $\theta$ ),  $\hat{y}$  + cos( $\theta$ ) · 1;
```

```
  r := 0  $\oplus_{0.5}$  1
```

```
od
```

(b) Forward-AD, dual number.

```
q1 := 0; q2 := 0; qc := 1;
```

```
r,  $\hat{r}$  := 0, 0; y,  $\hat{y}$  := 0, 0;
```

```
while r = 0 do
```

```
  if q1 = 0 then
```

```
    q2 := 0  $\oplus_{f(q_c)}$  1;
```

```
    if q2 = 0 then q1 := 1;
```

```
      y,  $\hat{y}$  := y + sin( $\theta$ ),  $\hat{y}$  + cos( $\theta$ ) · 1; (*)
```

```
    else qc := qc + 1 fi
```

```
  else y,  $\hat{y}$  := y + sin( $\theta$ ),  $\hat{y}$  + cos( $\theta$ ) · 0 fi;
```

```
  r := 0  $\oplus_{0.5}$  1
```

```
od
```

(c) EUL with only one dual number initialized.

Fig. 4. A probabilistic program $P(\theta)$ with forward-AD and our EUL-based AD applied, where $0 \oplus_p 1$ is a probabilistic choice that outputs 0 with probability p and 1 with probability $1 - p$ for any $0 \leq p \leq 1$, $f(n) = \mu(n)/(1 - \sum_{j=1}^{n-1} \mu(j))$ is determined by the distribution μ mentioned in Section 3.2.

Explanation of the Syntax. Statement **skip** does nothing and terminates immediately. Initialization statement $q := |0\rangle$ sets the quantum variable q to $|0\rangle$. Unitary transformation statement $\bar{q} := U[\bar{q}]$ means perform a unitary U on the quantum register $\bar{q}$. Statement $\bar{q} := e^{-i\theta\sigma}[\bar{q}]$ gives a special *parameterized* form of unitary—*density operator simulation*—with σ a density operator and θ selected from θ . Sequential composition $P_1(\theta); P_2(\theta)$ means first executes $P_1(\theta)$, and when $P_1(\theta)$ terminates, it executes $P_2(\theta)$. Quantum case statement **if** ($\square m \cdot M[\bar{q}] = m \rightarrow P_m(\theta)$) **fi**, where $\square m$ indicates case branching by the value of m , means performs a measurement $M = \{M_m\}$ on $\bar{q}$ and then a subprogram $P_m(\theta)$ will be performed upon the outcome m of the measurement. In quantum loop statement **while** $M[\bar{q}] = 1$ **do** $P(\theta)$ **od**, a binary measurement $M = \{M_0, M_1\}$ is performed; if the measurement outcome is 0, then the program terminates; otherwise, the program executes the loop body $P(\theta)$ and continues the loop, potentially for an arbitrary number of rounds.

Remark 4.2. We provide some remarks on the above syntax.

- We add a statement $\bar{q} := \sigma$ as a more general initialization that sets the state of the quantum register $\bar{q}$ to be a representable density operator σ , where the “representable” means the density operator can be generated by a short parameterized quantum **while**-program P without parameters and while-loop statement.
- We use $\bar{q} := e^{-i\theta\sigma}[\bar{q}]$ to describe a generally parameterized unitary applied on $\bar{q}$. For any unitary U , there is a Hermitian operator H such that $U = e^{-iH}$ and for any Hermitian operator H , e^{-iH} (the quantum simulation of Hamiltonian H) is also a unitary. The parameterization we have chosen, i.e., density operator simulation ($e^{-i\theta\sigma}$), can also express general Hamiltonian simulation.⁴

⁴For any $e^{-i\theta H}$, we define a density operator $\sigma_H = (H - \mu I)/\text{tr}(H - \mu I)$, where μ is the ground eigenvalue of H , I the identity, and $H \neq \mu I$. Then, $e^{-i\theta H}$ is the same as $e^{-i\theta'\sigma_H}$, where $\theta' = \text{tr}(H - \mu I)\theta$, since $e^{-i\theta H}\rho e^{i\theta H} = e^{-i(\text{tr}(H - \mu I)\theta)\sigma_H}\rho e^{i(\text{tr}(H - \mu I)\theta)\sigma_H}, \forall \rho$.

- Our parameterization allows expressing many commonly used parameterized quantum gates, such as Pauli rotation gates and two-qubit coupling gates, which are universal and can be reliably implemented in near-term quantum machines.⁵

4.2 Denotational Semantics

Following the semantics of quantum **while**-programs [73], the denotational semantics of parameterized quantum **while**-programs can be defined.

Definition 4.3 (Structural Representation of Denotational Semantics [73]). Let $\mathcal{H}_{all}$ denote the tensor product of the state spaces of all quantum variables and $\rho \in \mathcal{D}(\mathcal{H}_{all})$ indicate the (global) state of quantum variables. The denotational semantics of a parameterized quantum **while**-program $P(\theta)$ is a superoperator $\llbracket P(\theta) \rrbracket : \mathcal{D}(\mathcal{H}_{all}) \rightarrow \mathcal{D}(\mathcal{H}_{all})$ inductively defined as:

- $\llbracket \text{skip} \rrbracket (\rho) = \rho$;
- $\llbracket q := |0\rangle \rrbracket (\rho) = \sum_n |0\rangle_q \langle n| \rho |n\rangle_q \langle 0|$;
- $\llbracket \bar{q} := U[\bar{q}] \rrbracket (\rho) = U \rho U^\dagger$;
- $\llbracket \bar{q} := e^{-i\theta\sigma} [\bar{q}] \rrbracket (\rho) = e^{-i\theta\sigma} \rho e^{i\theta\sigma}$;
- $\llbracket P_1(\theta); P_2(\theta) \rrbracket (\rho) = \llbracket P_2(\theta) \rrbracket (\llbracket P_1(\theta) \rrbracket (\rho))$;
- $\llbracket \text{if } (\Box m \cdot M[\bar{q}] = m \rightarrow P_m(\theta)) \text{ fi} \rrbracket (\rho) = \sum_m \llbracket P_m(\theta) \rrbracket (\mathcal{E}_m(\rho))$;
- $\llbracket \text{while } M[\bar{q}] = 1 \text{ do } P(\theta) \text{ od} \rrbracket (\rho) = \bigsqcup_{n=0}^{\infty} \sum_{k=0}^n \mathcal{E}_0 \circ (\llbracket P(\theta) \rrbracket \circ \mathcal{E}_1)^k (\rho)$,

where $\{|n\rangle_q\}$ is an orthonormal basis of state space $\mathcal{H}_q$ of variable q , $\mathcal{E}_m : \rho \mapsto M_m \rho M_m^\dagger$ are defined for each measurement $M = \{M_m\}$ in $P(\theta)$, and $\bigsqcup$ stands for the least upper bound in the CPO of partial density operators with the Löwner order $\sqsubseteq$ (see Reference [73, Lemma 3.3.2]).

For a quantum program P , we define $\text{var}(P)$ to be the set of quantum variables $q \in \mathbf{qVar}$ appearing in a program P , and let $\mathcal{H}_P = \bigotimes_{q \in \text{var}(P)} \mathcal{H}_q$. When dealing with the semantics of a program $P(\theta)$, we only consider the states on $\mathcal{H}_{P(\theta)}$, that is, using $\rho \in \mathcal{H}_{P(\theta)}$ to represent a product state $\rho \otimes \rho_0 \in \mathcal{D}(\mathcal{H}_{all})$, where $\rho_0 \in \mathcal{D}(\mathcal{H}_{\mathbf{qVar} \setminus \text{var}(P(\theta))})$. When the dimension of $\mathcal{H}_{P(\theta)}$ is finite, we have that $\mathcal{D}(\mathcal{H}_{P(\theta)})$ is a compact set, thus

$$\llbracket \text{while } M[\bar{q}] = 1 \text{ do } P(\theta) \text{ od} \rrbracket (\rho) = \sum_{k=0}^{\infty} \mathcal{E}_0 \circ (\llbracket P(\theta) \rrbracket \circ \mathcal{E}_1)^k (\rho). \quad (4.1)$$

Note further that a semantic mapping $\llbracket P(\theta) \rrbracket : \mathcal{D}(\mathcal{H}_{all}) \rightarrow \mathcal{D}(\mathcal{H}_{all})$ defined on $\mathcal{D}(\mathcal{H}_{all})$ can be used as a mapping on the set of linear operators $\mathcal{L}(\mathcal{H}_{all})$ by linear extension.

4.3 Expectation Functions and Differentiability

The output of a quantum program is often regarded as the expectation of an observable obtained by measurements after its execution. We define the *expectation functions* to capture the output of quantum programs, which is similar to the observable semantics introduced by Reference [77].

Definition 4.4 (Expectation Function). For a parameterized quantum **while**-program $P(\theta)$ with parameter $\theta \in \mathbb{R}^k$, an initial state $\rho \in \mathcal{D}(\mathcal{H}_{all})$, and an observable O on $\mathcal{H}_{all}$, the expectation function $f : \mathbb{R}^k \rightarrow \mathbb{R} \cup \{\pm\infty\}$ that maps the parameter θ to the output expectation is defined by

$$f(\theta) = \text{tr}(O \llbracket P(\theta) \rrbracket (\rho)). \quad (4.2)$$

⁵Single-qubit Pauli rotation gates are given in the following form $R_\Delta(\theta) := \exp(-\frac{i\theta}{2}\Delta)$, $\Delta \in \{X, Y, Z\}$. One can also extend Pauli rotations to multiple qubits. For example, consider two-qubit coupling gates $\{R_{\Delta \otimes \Delta} := \exp(-\frac{i\theta}{2}\Delta \otimes \Delta)\}_{\Delta \in \{X, Y, Z\}}$. Note that these two-qubit gates can generate entanglement between two qubits. Combined with single-qubit rotations, they form a *universal gate set* for quantum computation.

For any $1 \leq j \leq k$, the partial derivatives $\frac{\partial f}{\partial \theta_j}$ of expectation function f with respect to parameter θ_j can be defined in the standard way. Their existence in the finite-dimensional case is guaranteed by the following:

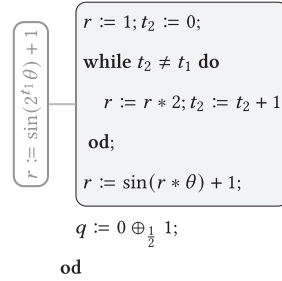
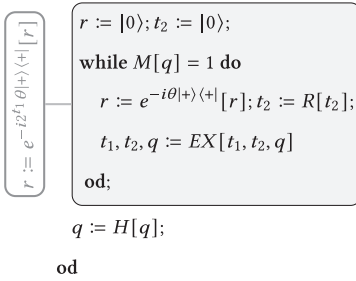
THEOREM 4.5 (DIFFERENTIABILITY). *For a parameterized quantum **while**-program $P(\theta)$ with parameter $\theta = (\theta_1, \theta_2, \dots, \theta_k) \in \mathbb{R}^k$, an initial state $\rho \in \mathcal{D}(\mathcal{H}_{all})$, and an observable O on $\mathcal{H}_{all}$, if $\mathcal{H}_{P(\theta)}$ is finite-dimensional, then $\frac{\partial f}{\partial \theta_i}$ exists.*

PROOF. This is a corollary of Theorem 5.4, which states that $\frac{\partial f}{\partial \theta_i}$ can be represented by the expectation function of another quantum program, its differential program $\frac{\partial}{\partial \theta_i}(P(\theta))$, with respect to an observable $O_d \otimes O$. Since the $O_d \otimes O$ in Theorem 5.4 is bounded, the expectation function of $\frac{\partial}{\partial \theta_i}(P(\theta))$ with respect to any θ , observable $O_d \otimes O$, and any input state ρ is well-defined, we have that $\frac{\partial f}{\partial \theta_i}$ is well-defined and hence exists. $\square$

As shown in the following example, however, it is possible that the expectation function f is non-differentiable when $\mathcal{H}_{P(\theta)}$ is an infinite-dimensional space.

Example 4.6 (Non-differentiable Infinite-dimensional Quantum Program). Let q, r be two qubits with state space $\mathcal{H}_q = \mathcal{H}_r = \text{span}\{|0\rangle, |1\rangle\}$, t_1, t_2 be quantum variables with state space $\mathcal{H}_\infty = \{\sum_{n=-\infty}^{\infty} \alpha_n |n\rangle : \alpha_n \in \mathbb{C}, \sum_n |\alpha_n|^2 < \infty\}$, and $\theta \in \mathbb{R}$ be a parameter. Consider the following parameterized quantum program $P(\theta)$:

$$\begin{aligned} P(\theta) \equiv & q := |0\rangle; t_1 := |0\rangle; \\ & \textbf{while } M[q] = 1 \textbf{ do} \\ & \quad t_1 := R[t_1]; \end{aligned}$$

$$\begin{aligned} C(\theta) \equiv & q := 0; t_1 := 0; \\ & \textbf{while } q \neq 1 \textbf{ do} \\ & \quad t_1 := t_1 + 1; \end{aligned}$$


where:

- $M = \{M_0 = |1\rangle\langle 1|, M_1 = |0\rangle\langle 0|\}$ is the measurement on qubit q in the computational basis;
- $R = \sum_j |j+1\rangle\langle j|$ is the right-translation operator on t_1 ; and
- $EX = \sum_{2^j=k} |jk\rangle\langle jk| \otimes X + \sum_{2^j \neq k} |jk\rangle\langle jk| \otimes I$ is a unitary that performs X operation on q if t_1 and t_2 is in state $|j\rangle$ and $|k\rangle$, respectively, and $k = 2^j$ for any $j, k \in \mathbb{Z}$.

For an initial state $\rho = |0\rangle_q \langle 0| \otimes |0\rangle_r \langle 0| \otimes |0\rangle_{t_1} \langle 0| \otimes |0\rangle_{t_2} \langle 0|$ and an observable $O = 2|\psi\rangle_r \langle \psi|$ with $|\psi\rangle = (|0\rangle - i|1\rangle)/\sqrt{2}$, a calculation using Equation (4.1) yields the expectation function of $P(\theta)$:

$$f(\theta) = \sum_{k=1}^{\infty} \frac{1}{2^k} \text{tr} \left(2|\psi\rangle_r \langle \psi| e^{-i2^k \theta |+\rangle_r \langle +|} |0\rangle_r \langle 0| e^{i2^k \theta |+\rangle_r \langle +|} \right) = \sum_{k=1}^{\infty} \frac{1 + \sin(2^k \theta)}{2^k} = 1 + \sum_{k=1}^{\infty} \frac{1}{2^k} \sin(2^k \theta),$$

which is well-defined. However, f is non-differentiable everywhere due to Weierstrass's non-differentiable function [33, Theorem 1.31]: The function $S(x) = \sum_{n=0}^{\infty} a^n \sin(b^n x)$ converges

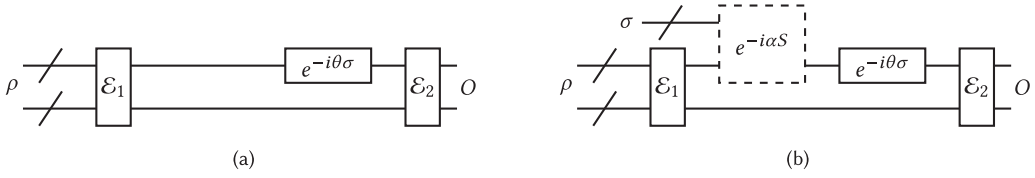


Fig. 5. Introduce the commutator with a similar circuit.

uniformly on $\mathbb{R}$, which implies S is continuous on $\mathbb{R}$, but nowhere differentiable for any $0 < a < 1$, $ab \geq 1$.

The probabilistic program $C(\theta)$ is a counterpart of $P(\theta)$ for illustration, where $q := 0 \oplus_{\frac{1}{2}} 1$ assigns 0 to q with probability $\frac{1}{2}$ and 1 to q otherwise. The boxed commands assigns $\sin(2^{t_1}\theta) + 1$ to r , thus, we can see that the expectation of variable r after runs $C(\theta)$ is also $f(\theta)$ and non-differentiable everywhere.

5 AUTOMATIC DIFFERENTIATION FOR UNBOUNDED QUANTUM LOOPS

In this section, we develop the AD technique for parameterized quantum **while**-programs to overcome the major difficulty of finding analytical derivatives of unbounded loops.

5.1 Differentiation on a Single-occurrence of Parameter

Our first contribution is a new DSOP technique, called the commutator-form rule. Li et al. [42] and Mitarai et al. [49] first proposed a derivative formula for Pauli rotations, which is named by Schuld et al. [59] as the “parameter-shift rule” to handle the case of $U(\theta) = e^{-i\theta H}$ with H having at most two distinct eigenvalues.

Our commutator-form rule is designed to be applicable to $e^{-i\theta H}$ for general H . Technically, it was inspired by a few existing works [9, 44, 49] that leverage the commutator form for various purposes. We also note some recent independent developments [35, 39, 68] of variants of the parameter-shift rules to handle more general $e^{-i\theta H}$. However, our rule is based on a very different technique, which could be of independent interest by itself. Precisely,

LEMMA 5.1. *Let $\mathcal{H}_1, \mathcal{H}_2, \mathcal{H}_3$ be Hilbert spaces and $\mathcal{E}_1 : \mathcal{D}(\mathcal{H}_1) \rightarrow \mathcal{D}(\mathcal{H}_2)$, $\mathcal{E}_2 : \mathcal{D}(\mathcal{H}_2) \rightarrow \mathcal{D}(\mathcal{H}_3)$ be superoperators. For any Hermitian operator H on $\mathcal{H}_2$ and $\theta \in \mathbb{R}$, we define $\mathcal{E}_{H,\theta}(\rho) = e^{-i\theta H} \rho e^{i\theta H}$ for all $\rho \in \mathcal{D}(\mathcal{H}_2)$. Then, for any density operator ρ on $\mathcal{H}_1$:*

$$\frac{d}{d\theta} (\mathcal{E}_2 \circ \mathcal{E}_{H,\theta} \circ \mathcal{E}_1(\rho)) = \mathcal{E}_2 \circ \mathcal{E}_{H,\theta}(-i[H, \mathcal{E}_1(\rho)]),$$

where commutator $[\cdot, \cdot]$ is defined as follows: $[A, B] = AB - BA$ for any operators A and B .

PROOF. See Appendix D.2. □

Commutator-form Rule. The way of introducing commutators is visualized in Figure 5. We define

$$f(\theta) = \text{tr}(O \mathcal{E}_2(e^{-i\theta\sigma} \mathcal{E}_1(\rho) e^{i\theta\sigma})) \quad (5.1)$$

as the expectation function in Figure 5(a) and

$$g(\theta; \alpha) = \text{tr}(O \mathcal{E}_2(e^{-i\theta\sigma} e^{-i\alpha S} \mathcal{E}_1(\rho) \otimes \sigma e^{i\alpha S} e^{i\theta\sigma})) \quad (5.2)$$

$$\begin{aligned}
T_\theta(\text{skip}) &\equiv \text{skip} & T_\theta(q := |0\rangle) &\equiv q := |0\rangle & T_\theta(\bar{q} := U[\bar{q}]) &\equiv \bar{q} := U[\bar{q}] \\
T_\theta(\bar{q} := e^{-i\theta'\sigma}[\bar{q}]) &\equiv \bar{q} := e^{-i\theta'\sigma}[\bar{q}] & (\theta' \neq \theta) \\
T_\theta(P_1(\theta); P_2(\theta)) &\equiv T_\theta(P_1(\theta)); T_\theta(P_2(\theta)) \\
T_\theta(\text{if } (\Box m \cdot M[\bar{q}] = m \rightarrow P_m(\theta)) \text{ fi}) &\equiv \text{if } (\Box m \cdot M[\bar{q}] = m \rightarrow T_\theta(P_m(\theta))) \text{ fi} \\
T_\theta(\text{while } M[\bar{q}] = 1 \text{ do } P(\theta) \text{ od}) &\equiv \text{while } M[\bar{q}] = 1 \text{ do } T_\theta(P(\theta)) \text{ od} \\
(*) \quad T_\theta(\bar{q} := e^{-i\theta\sigma}[\bar{q}]) &\equiv \text{if } (M_{q_1, q_2}[q_1, q_2] = 0 \rightarrow q_c := C[q_c]; q_c, q_2 := GP[q_c, q_2] \\
&\quad \Box = 1 \rightarrow q_1 := X[q_1]; q_2 := \frac{1}{2}; \bar{q}' := \sigma; \\
&\quad \quad q_2, \bar{q}, \bar{q}' := AS[q_2, \bar{q}, \bar{q}'] \\
&\quad \Box = 2 \rightarrow \text{skip}) \text{ fi}; \bar{q} := e^{-i\theta\sigma}[\bar{q}]
\end{aligned}$$

Fig. 6. Code transformation rules with respect to parameter θ . The blue part of $(*)$ refers to the EUL part of Figure 3, where q_1, q_2 are two qubit variables, q_c is a quantum variable with state space $\mathcal{H}_c = \text{span}\{|n\rangle : n \in \mathbb{Z}\}$, $M_{q_1, q_2} = \{M_0 = |00\rangle\langle 00|, M_1 = |01\rangle\langle 01|, M_2 = |10\rangle\langle 10| + |11\rangle\langle 11|\}$, $C = \sum_{j=0}^{\infty} |j+1\rangle\langle j|$ is the right-translation operator, $GP = \sum_{j=1}^{\infty} |j\rangle\langle j| \otimes R_y(2 \arcsin(\sqrt{b_j}))$ and $b_j = \mu(j)/(1 - \sum_{k=1}^{j-1} \mu(k))$, $AS = |0\rangle\langle 0| \otimes e^{-i\frac{\pi}{4}S_{\bar{q}, \bar{q}'}} + |1\rangle\langle 1| \otimes e^{i\frac{\pi}{4}S_{\bar{q}, \bar{q}'}}$ and $S_{\bar{q}, \bar{q}'}$ is the SWAP operator between $\mathcal{H}_{\bar{q}}$ and $\mathcal{H}_{\bar{q}'}$.

as the expectation function in Figure 5(b), where S is the SWAP operator.⁶ With Lemma 5.1, we have

$$\frac{d}{d\theta} f(\theta) = \text{tr} \left(O\mathcal{E}_2 \left(e^{-i\theta\sigma} (-i[\sigma \otimes I, \mathcal{E}_1(\rho)]) e^{i\theta\sigma} \right) \right).$$

Inspired by the trick of applying unitary transformation $e^{-i\theta\rho}$ of any density operator ρ in quantum principal component analysis [44], we find that for any $\alpha \in (0, \frac{\pi}{2})$:

$$(\text{commutator-form rule}) \quad \frac{d}{d\theta} f(\theta) = \frac{1}{\sin(2\alpha)} (g(\theta; \alpha) - g(\theta; -\alpha)). \quad (5.3)$$

5.2 Code Transformation for Unbounded Loops

Our AD scheme (Figure 3) could leverage any DSOP technique (both the commutator-form rule and the parameter-shift rule). We illustrate the code-transformation based on the commutator-form rule and leave the details based on the parameter-shift rule in Appendix B.

Definition 5.2 (Code Transformation). For a parameterized quantum **while**-program $P(\theta)$ with parameter $\theta \in \mathbb{R}^k$, its differential program with respect to θ is defined as a parameterized quantum **while**-program $\frac{\partial}{\partial\theta}(P(\theta))$:

$$\frac{\partial}{\partial\theta}(P(\theta)) \equiv \text{Dinit}; T_\theta(P(\theta)),$$

with **Dinit** defined as follows and T_θ, C, GP given in Figure 6,

$$\text{Dinit} \equiv q_1 := |0\rangle; q_2 := |0\rangle; q_c := |0\rangle; q_c := C[q_c]; q_c, q_2 := GP[q_c, q_2].$$

The code transformation T_θ in Figure 6 only acts non-trivially for unitary transformation statements that contain the parameter θ , that is, inserting a measurement statement (the blue part of rule $(*)$ in Figure 6) before a parameterized unitary transformation $\bar{q} := e^{-i\theta\sigma}[\bar{q}]$. This measurement statement corresponds to the EUL commands of our AD scheme in Figure 3.

⁶The SWAP operator S on a space $\mathcal{H} \otimes \mathcal{H}$ is defined as $S(|a\rangle \otimes |b\rangle) = |b\rangle \otimes |a\rangle$ for any $|a\rangle, |b\rangle \in \mathcal{H}$ that swaps the states of two systems.

To establish the correctness of our code transformation, we develop the following lemma about finite-dimensional quantum programs in light of Example 4.6:

LEMMA 5.3. *Consider a quantum loop $P \equiv \text{while } M[\bar{q}] = 1 \text{ do } Q \text{ od}$. Assume that the state space $\mathcal{H}_P$ is finite-dimensional. We define superoperators $\mathcal{E}_i : \mathcal{D}(\mathcal{H}_P) \rightarrow \mathcal{D}(\mathcal{H}_P)$ by $\mathcal{E}_i(\rho) = M_i \rho M_i^\dagger$, $i = 0, 1$ and $\mathcal{E} : \mathcal{D}(\mathcal{H}_P) \rightarrow \mathcal{D}(\mathcal{H}_P)$ by $\mathcal{E}(\rho) = \llbracket Q \rrbracket(\rho)$. Then, for any $\epsilon \in (0, 1)$, there exists $N = N_\epsilon > 0$ such that $\forall n \in \mathbb{N}, \forall \rho \in \mathcal{D}(\mathcal{H}_P)$,*

$$\text{tr}(\mathcal{E}_0 \circ (\mathcal{E} \circ \mathcal{E}_1)^n(\rho)) \leq \epsilon^{\lfloor \frac{n}{N} \rfloor} \text{tr}(\rho).$$

PROOF. See Appendix D.1. □

The above lemma ensures the probability that the finite-dimensional program runs out of the loop has an exponential decay on the number of loop iterations. This observation leads to an exponential decay of partial derivatives for corresponding occurrences of the parameter, which in turn guarantees the existence of the derivative and the validity of exchanging the order between the infinite summation and the derivation.

THEOREM 5.4 (CORRECTNESS OF CODE TRANSFORMATION). *Given a parameterized quantum **while**-program $P(\theta)$ with parameter $\theta \in \mathbb{R}^k$ and finite-dimensional state space $\mathcal{H}_{P(\theta)}$, an observable O , and an input state ρ . Let $f(\theta)$ be the expectation function of $P(\theta)$ with respect to ρ and O . Then, the partial derivative of f with respect to θ is*

$$\frac{\partial}{\partial \theta} f(\theta) = \text{tr} \left((O_d \otimes O) \left[\left[\frac{\partial}{\partial \theta} (P(\theta)) \right] (\rho) \right) \right), \quad (5.4)$$

the expectation function of $\frac{\partial}{\partial \theta} (P(\theta))$ with respect to θ , observable $O_d \otimes O$ and input state ρ with $O_d = \sum_{j=1}^{\infty} \frac{2}{\mu(j)} |j\rangle\langle j| \otimes |1\rangle\langle 1| \otimes Z$ is an observable on $\mathcal{H}_{q_c} \otimes \mathcal{H}_{q_1} \otimes \mathcal{H}_{q_2}$.

OUTLINE OF THE PROOF. We can take Figure 3 as an example to briefly illustrate the outline of the proof, while the full details are deferred to Appendix D.4.

- (1) Since our AD is performed by inserting commands, the execution branches of $P(\theta)$'s differential program in Figure 3(c) are the same as $P(\theta)$ in Figure 3(a). Thus, we consider each execution path of $P(\theta)$.
- (2) For a fixed execution path of $P(\theta)$, its derivative has the form $\partial_\theta f_k(\theta) \equiv \frac{1}{2^{k+1}} \sum_{j=1}^k (g_\theta)^{j-1} \circ (\partial_\theta g_\theta) \circ (g_\theta)^{k-j}$. The $\partial_\theta f_k(\theta)$ corresponds to perform k times differentiation operations in different occurrences of θ . For the same branch of the fixed execution path, $P(\theta)$'s differential program can also perform the same k times differentiation operations with probability $\mu(1), \dots, \mu(k)$. Then, $P(\theta)$'s differential program can produce $\partial_\theta f_k(\theta)$ by estimation of expectation.
- (3) Finally, one adds up all the $\partial_\theta f_k(\theta)$ that are produced by $P(\theta)$'s differential program with respect to $P(\theta)$'s execution paths: $\sum_{k=1}^{\infty} \partial_\theta f_k(\theta)$, and prove it is uniformly convergent, the main challenging of the proof that relies on the finite-dimensional condition, and equal to $f(\theta)$'s derivative.

□

6 IMPLEMENTATION AND SAMPLE COMPLEXITY

In this section, we discuss the implementation of our AD scheme and analyze its efficiency in terms of sample complexity, the number of required samples to estimate gradients.

6.1 Implementation in a Hybrid Style

In the previous section, we have constructed the code transformation for AD in a pure quantum fashion so differential programs can be written in the same syntax as the original one. But this approach introduces three additional quantum variables on top of the original program, which requires additional quantum resources. Since the cost of quantum hardware implementation is still very high, to make our AD more practical, we need to find ways to reduce additional quantum resources. Fortunately, our construction T_θ in Figure 6 guarantees that no entanglement will be created between q_1, q_2, q_c and other quantum variables in the differential program, which means that there is only classical correlation rather than quantum correlation (see Reference [??, Section VI. Bipartite Entanglement]) between q_1, q_2, q_c and other quantum variables. Moreover, q_c will always be in its basis states $\{|n\rangle : n \in \mathbb{Z}\}$ and q_1, q_2 are two qubit variables. Therefore, q_1, q_2, q_c can be separated from the differential program and simulated efficiently by a classical computer. As a result, our AD can be implemented in languages that support both quantum and classical operations, which refers to hybrid quantum-classical programming.

There are a few candidates of high-level quantum programming languages that support hybrid quantum-classical programming with classical control flow, e.g., Microsoft's Q# [62] and ETH Zürich's Silq [13]. Since Q# provides a Python package qsharp that enables simulation of Q# programs from regular Python programs, we choose Python and Q# to implement a parser that transforms parameterized quantum **while**-programs (with a restricted set of unitaries) to Q# and implement AD to generate Q# codes for evaluating gradients.

Our current implementation supports parameterized Pauli rotations and controlled Pauli rotations as follows:

$$\left\{ R_\Delta(\theta) = e^{-i\frac{\theta}{2}\Delta}, e^{-i\frac{\theta}{2}|1\rangle\langle 1| \otimes \Delta}, R_{\Delta \otimes \Delta}(\theta) = e^{-i\frac{\theta}{2}\Delta \otimes \Delta} \quad : \Delta = X, Y, Z; \theta \in \theta \right\}.$$

The Pauli rotations and controlled Pauli rotations are internally replaced with their corresponding density operator form, e.g., unitary $e^{-i\frac{\theta}{2}X \otimes X}$ is replaced by $e^{-i(2\theta)\frac{X \otimes X + I}{4}}$ with $(X \otimes X + I)/4$, a density operator, then we can apply our technique of AD to it and get the derivative with a scale 2.

6.2 Variance and Sample Complexity

Our main theorem (Theorem 5.4) asserts that the desired partial derivative can be expressed by the expectation of observable $O_d \otimes O$ with respect to state $\llbracket \frac{\partial}{\partial \theta}(P(\theta)) \rrbracket(\rho)$, which we denote $\langle O_d \otimes O \rangle$ for simplicity. We denote the *sample complexity* as the number of repetitions to estimate $\langle O_d \otimes O \rangle$ to a given precision δ . To estimate the sample complexity, we consider the *variance* of observable $O_d \otimes O$: $\text{Var}(O_d \otimes O) = \langle (O_d \otimes O - \langle O_d \otimes O \rangle)^2 \rangle = \langle O_d^2 \otimes O^2 \rangle - \langle O_d \otimes O \rangle^2$.

Inspired by the “Occurrence Count for θ ” in Reference [77], we introduce two technical notions, i.e., the “Running Count for θ ” in program $P(\theta)$, denoted $RC_\theta(P(\theta))$, as the number of occurrences of θ in $P(\theta)$, and the “Loop Count” in $P(\theta)$, denoted $LC(P(\theta))$, as the number of while-loop statements in $P(\theta)$, for upper bounding $\langle O_d^2 \otimes O^2 \rangle$. For formal definitions of $RC_\theta(P(\theta))$ and $LC(P(\theta))$, please refer to Appendix A. We also need a terminating condition of parameterized programs to upper bound $\langle O_d^2 \otimes O^2 \rangle$.

Definition 6.1 (Almost Sure Termination [73]). A parameterized quantum **while**-program $P(\theta)$ terminates almost surely at θ if $\text{tr}(\llbracket P(\theta) \rrbracket(\rho)) = \text{tr}(\rho)$ for any $\rho \in \mathcal{D}(\mathcal{H}_{P(\theta)})$.

THEOREM 6.2. *In the same setting as in Theorem 5.4 and distribution $\mu : \mathbb{Z}_+ \rightarrow [0, 1]$ satisfies [converging-rate condition](#), if all the **while**-statements (subprograms) in $P(\theta)$ terminate almost surely,*

then $\langle O_d^2 \otimes O^2 \rangle$ is bounded. Additionally, if the distribution $\mu : \mathbb{Z}_+ \rightarrow [0, 1]$ satisfies

$$\mu(n) \propto \frac{1}{n \ln^{1+s}(n+e)} \text{ with constant } s \in (0, 1], \quad (6.1)$$

then we have

$$\langle O_d^2 \otimes O^2 \rangle \in O \left(M_1^2 \ln^{1+s}(M_1 + e) + M_1^{3+s} M_2^{2+s} C(M_2) \right)$$

with $M_1 = RC_\theta(P(\theta))$, $M_2 = LC(P(\theta))$, and $C(M_2)$ is a non-zero function of M_2 .

PROOF. See Appendix A. □

Comparison with Zhu et al. [77]. In the case of no unbounded loops in Reference [77], we have $M_2 = 0$ and the bound given in Theorem 6.2 becomes $O(M_1^2 \ln^{1+s}(M_1 + e))$, which implies the sample complexity $O(M_1^2 \ln^{1+s}(M_1 + e)/\delta^2)$ by Chebyshev's Inequality. This is comparable to the sample complexity $O(m^2/\delta^2)$ estimated in Reference [77], where $m \approx RC_\theta(P(\theta)) = M_1$, as all of the loops considered there are bounded and thus can be unfolded to nested conditional statements.

Empirical Estimation of the Sample Bound. The bound in Theorem 6.2 could, however, be loose in practice, which would cost unnecessary samples. To resolve this issue, we develop an *empirical* estimation of the sample bound, which usually leads to tighter bounds in our case studies.

Our key idea is that one can empirically estimate $\langle O_d^2 \otimes O^2 \rangle$ by sampling as we did for $\langle O_d \otimes O \rangle$ to get a better empirical bound than analytical ones. To that end, one can apply a similar technique in Theorem 6.2 to bound $\langle O_d^4 \otimes O^4 \rangle$ and hence the number of samples required to estimate $\langle O_d^2 \otimes O^2 \rangle$. However, at this time, we can tolerate a much larger additive error δ , since $\langle O_d^2 \otimes O^2 \rangle$ could be large itself, which makes $1/\delta^2$ in Chebyshev's Inequality scale nicely.

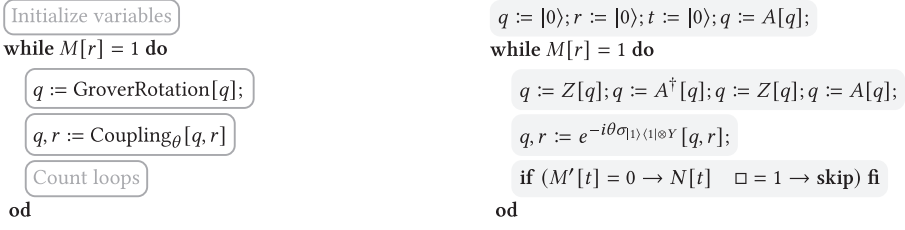
7 CASE STUDIES

In this section, we present the case studies to demonstrate the feasibility of our framework, including *parameterized amplitude amplification*, *quantum walk-based search algorithm*, and *repeat-until-success unitary implementation*. The chosen case studies, all of which contain unbounded quantum loops, are non-trivial and realistic examples from quantum literature. We do not choose typical variational algorithms, e.g., QAOA [22], VQE [54], or some variants studied in the previous work of differentiable quantum programming [77], since they do not contain unbounded loops. Similarly, because there is no realistic example yet of nested loops, as existing quantum algorithms are far less than classical, we do not artificially construct experiments for nested loops. However, our proposed commutator-form rule provides a more concise form than the parameter-shift rule for general Hamiltonian (e.g., Hamiltonian in QAOA [32]) and our inductively defined code transformation can handle nested loops.

Experiment Workflow. For experiments, our framework provides a unified principled way to identify suitable parameters of parameterized quantum **while**-programs automatically as follows:
Given: A parameterized quantum **while**-program $P(\theta)$, $\theta \in \mathbb{R}^k$, $k \geq 1$, a quantum state ρ as program's input, and an observable O defined on $\mathcal{H}_{P(\theta)}$.

Workflow:

- (1) Use the implemented parser in Section 6.1 to convert the program $P(\theta)$ to $Q\#$ functions that can sample the value and the partial derivatives of expectation function $f(\theta) = \text{tr}(O \llbracket P(\theta) \rrbracket(\rho))$, which is the objective function to optimize.
- (2) Use the empirical estimation of the sample bound developed in Section 6.2 to estimate the number of samples needed for sampling the partial derivatives of $f(\theta)$.



(a) Overview of parameterized AA in Fig. 1.

(b) Specific instance of parameterized AA: $P_1(\theta)$.

Fig. 7. Parameterized AA used in the experiment.

- (3) Use a gradient-based optimizer (in our experiments, we choose Adam optimizer [37], as it is widely used; some other optimizers are also suitable, e.g., AdamW [45]) to *maximize/minimize* $f(\theta)$, where the initial value of parameters θ_0 is usually randomly given and the gradient of $f(\theta)$ is estimated by the Q# functions (all run on the simulator provided by Q#) in (1) with the number of samples estimated in (2).

In all experiments, the distribution μ in code transformation for AD is chosen as the distribution in Equation (6.1) with $s = 0.25$. Our experiments are performed on a desktop computer with Intel(R) Core(TM) i7-9700 CPU @ 3.00 GHz $\times$ 8 Processors and 16 GB RAM.

7.1 Parameterized Amplitude Amplification

Amplitude amplification (AA) [15] is a generalization of Grover’s quantum search algorithm [29]. It employs an oracle unitary A , where $A|0\rangle = \sum_x \alpha_x |x\rangle$ (i.e., a superposition of all elements of a finite set), as well as the inverse $A^\dagger$ of A . Suppose the probability of obtaining target elements is p when performing measurement $\{M_x = |x\rangle\langle x|\}$ on state $A|0\rangle$. AA can find a target element by using $O(1/\sqrt{p})$ calls of A and $A^\dagger$.

An AA algorithm needs to run a specific number of a rotation operator without any intermediate measurement. Otherwise, the quantum speedup of oracle calls over classical algorithms may be lost. A novel idea, called the critically damped quantum search [50], challenged this phenomenon. It implemented a while-loop variant of Grover’s algorithm with a damping value, which has a critical value that divides between the quantum $O(1/\sqrt{p})$ and classical $O(1/p)$ search regimes. This critically damped quantum search can also be elegantly reformulated in a general framework that uses the while-loop primitive with a notion of κ -measurement [4]. With this framework for while-loop, the key issue is to find an appropriate value of κ to achieve the quantum speedup.

As the first case study, we show that our framework can be used to obtain a better parameter in the example of parameterized AA as in Figure 1 to not only obtain quantum speedup but also make fewer oracle calls than those given in the existing literature [4, 50] analytically by hand.

Parameterized AA Program. Consider the parameterized AA in a single-qubit system. Given $p \in (0, 1)$, suppose we have a single qubit unitary A such that $A|0\rangle = \sqrt{1-p}|0\rangle + \sqrt{p}|1\rangle$, and its inverse $A^\dagger$. State $|1\rangle$ is our target state. The details of parameterized AA program are listed in Figure 7, where we put the overview of parameterized AA and its instance $P_1(\theta)$ used in this experiment together.

In Figure 7(b), q, r are qubit variables, measurement $M = \{M_0 = |1\rangle\langle 1|, M_1 = |0\rangle\langle 0|\}$, and $\sigma_{|1\rangle\langle 1| \otimes Y} = (|1\rangle\langle 1| \otimes Y + I \otimes I)/4$. The variable r together with the unitary $e^{-i\theta\sigma_{|1\rangle\langle 1| \otimes Y}}$ and measurement M forms a $\frac{\theta}{4}$ -measurement in Reference [4]. To count the calls of A and $A^\dagger$ (the running number of loops), we introduce a block of “count loops” that does not affect the behavior of parameterized AA, where t is a quantum variable in the space $\mathcal{H}_p = \text{span}\{|0\rangle, \dots, |4\lfloor 1/\sqrt{p} \rfloor\rangle\}$, unitary

Table 3. Experiment Results on Parameterized AA

p	Ours		B		C	
	$\langle 4O_1 \rangle$	$\text{Var}(4O_1)$	$\langle 4O_1 \rangle$	$\text{Var}(4O_1)$	$\langle 4O_1 \rangle$	$\text{Var}(4O_1)$
$1/10^2$	0.6499	0.1184	1.6884	1.6480	0.6773	0.1872
$1/15^2$	0.6733	0.1232	1.7404	1.6336	0.7018	0.1872
$1/20^2$	0.6885	0.1216	1.7541	1.6752	0.7182	0.1904
$1/25^2$	0.6926	0.1200	1.7622	1.7504	0.7245	0.1936
$1/30^2$	0.6934	0.1152	1.7742	1.6624	0.7253	0.2112

The smaller $\langle 4O_1 \rangle$, the better query complexity.

$N = \sum_{n=0}^{4\lceil 1/\sqrt{p} \rceil - 1} |n+1\rangle_t \langle n| + |0\rangle_t \langle 4\lceil 1/\sqrt{p} \rceil|$, and measurement $M' = \{M'_0 = \sum_{n=0}^{4\lceil 1/\sqrt{p} \rceil - 1} |n\rangle_t \langle n|, M'_1 = I - M'_0\}$. With the conditional statement of measurement M' , the variable t will remain unchanged once it reaches the state $|4\lceil 1/\sqrt{p} \rceil\rangle$.

The program's input can be arbitrary, since there are variables' initialization in Figure 7(b). The observable we choose is $O_1 = \frac{\sqrt{p}}{4} \sum_{n=1}^{4\lceil 1/\sqrt{p} \rceil} n|n\rangle_t \langle n|$, which expresses the running number of loop iterations and represents the total oracle calls.⁷ The scale $\sqrt{p}/4$ is used to normalize the output of O_1 . We expect the oracle calls to be as few as possible, thus our **target** is to *identify θ so the expectation of the running time (O_1) of parameterized AA in Figure 7(b) is minimized.*

We summarize below the needed configuration in the experiment workflow.

Given: Parameterized AA program $P_1(\theta)$, arbitrary input state ρ , and observable O_1 .

Workflow: In (2), samples' number: $5/\sqrt{p} \times 10^3$. In (3), Adam's setting: $\beta_1 = 0.9, \beta_2 = 0.999, \alpha = 0.1$; initial parameter: $\theta = 4 \arccos((1 - 2\sqrt{p(1-p)})/(1 + 2\sqrt{p(1-p)}))$ (analytical but sub-optimal value from Reference [50]); Goal: **minimize** the expectation function of observable O_1 .

For the number of samples, a numerical calculation based on a finer version of Theorem 6.2 provides 799.72 as the bound of $\langle O_d^2 \otimes O_1^2 \rangle$ with $p = 1/100$. However, applying our empirical estimation, the actual value of $\langle O_d^2 \otimes O_1^2 \rangle$ would be bounded by 44.26 when $p = 1/100$, which leads to the current $5/\sqrt{p} \times 10^3$ bound ($= 5 \times 10^4$ when $p = 1/100$) with additive error $\delta = 0.1$ by Chebyshev's Inequality. Please refer to details in Appendix C.

Results. We choose $p = 1/10^2, 1/15^2, \dots, 1/30^2$ to run this experiment. In Table 3, we list the value of $\langle 4O_1 \rangle = 4\langle O_1 \rangle$, which expresses the (approximate) ratio of the number of loops to $1/\sqrt{p}$, that we find in this experiment (see the column "Ours"), as well as those in previous works [4] (see the column "B"), and Reference [50] (see the column "C") for the probability p specified in each row. For each p , a better result (both smaller $\langle 4O_1 \rangle$ and smaller variance $\text{Var}(4O_1)$ that implies less fluctuation around the expectation) is found by our experiment. Recall that both B and C results are based on analytical forms developed by domain experts.

Since our goal is to minimize the expectation of O_1 , we find that the experimental results confirm our framework's feasibility and validate the experiment workflow for automatically getting suitable parameters.

7.2 Quantum Walk with Parameterized Shift Operator

Quantum walk (QW) algorithms [2, 17, 64, 70], which share some similarities with Grover's algorithm, are vibrant in the area of quantum algorithms. In the context of the grid search, Benioff

⁷This is only an approximation of the total running time, since the state of t will always be $|4\lceil 1/\sqrt{p} \rceil\rangle$ after $4\lceil 1/\sqrt{p} \rceil$ loop iterations. But this does not matter, because the running number of loop iterations is concentrated below $1/\sqrt{p}$.

[11] observed that the standard Grover's search algorithm needs $\Omega(N)$ steps to find a marked vertex in an $\sqrt{N} \times \sqrt{N}$ grid. Quantum walk with a natural ("moving") shift operator S_m , which keeps the direction (also called the coin) after every move, also takes at least $\Omega(N)$ steps to find a marked vertex in this grid [3].

$$\begin{array}{ll}
 S_f : |\Leftarrow, x, y\rangle \rightarrow |\Rightarrow, x-1, y\rangle & S_m : |\Leftarrow, x, y\rangle \rightarrow |\Leftarrow, x-1, y\rangle \\
 |\Rightarrow, x, y\rangle \rightarrow |\Leftarrow, x+1, y\rangle & |\Rightarrow, x, y\rangle \rightarrow |\Rightarrow, x+1, y\rangle \\
 |\Uparrow, x, y\rangle \rightarrow |\Downarrow, x, y+1\rangle & |\Uparrow, x, y\rangle \rightarrow |\Uparrow, x, y+1\rangle \\
 |\Downarrow, x, y\rangle \rightarrow |\Uparrow, x, y-1\rangle & |\Downarrow, x, y\rangle \rightarrow |\Downarrow, x, y-1\rangle
 \end{array}$$

They resolved this issue by introducing another shift operator S_f , which can be interpreted as changing direction after every move, and the quantum walk associated with S_f takes $O(\sqrt{N} \log N)$ steps to find a marked vertex in this grid.

We can see that designing a shift operator, the direction (coin) transformation, is important for the performance of the quantum walk. It motivates us to parameterize the shift operator and use our framework to determine a good shift operator.

Parameterized QW Program. The quantum walk search algorithm in Reference [3] first initializes the coin variable and position variable in the uniform superposition and applies the marked quantum walk operator for several times (here, we apply it twice), then measure the position variable to check if the measured vertex is the marked one. We parameterize the shift operator and write it as follows:

$$\begin{aligned}
 P_2(\theta_1, \theta_2) &\equiv t := |0\rangle; \\
 &\textbf{while } M[q_x, q_y] = 1 \textbf{ do} \\
 &\quad c_x := |0\rangle; c_y := |0\rangle; q_x := |0\rangle; q_y := |0\rangle; \\
 &\quad c_x := H[c_x]; c_y := H[c_y]; q_x, q_y := \tilde{H}[q_x, q_y]; \\
 &\quad c_x, c_y, q_x, q_y := C[c_x, c_y, q_x, q_y]; c_x, c_y, q_x, q_y := S(\theta_1, \theta_2)[c_x, c_y, q_x, q_y]; \\
 &\quad c_x, c_y, q_x, q_y := C[c_x, c_y, q_x, q_y]; c_x, c_y, q_x, q_y := S(\theta_1, \theta_2)[c_x, c_y, q_x, q_y]; \\
 &\quad \textbf{if } (M'[t] = 0 \rightarrow A[t] \quad \square = 1 \rightarrow \textbf{skip}) \textbf{ fi od},
 \end{aligned} \tag{7.1}$$

where c_x and c_y are two qubit variables for coin, indicating the directions $\Leftarrow, \Rightarrow$ and $\Uparrow, \Downarrow$, respectively. q_x, q_y are two variables with space $\mathcal{H}_{\sqrt{N}} = \{|0\rangle, \dots, |\sqrt{N}-1\rangle\}$, indicating the position. The variable t is a variable with space $\mathcal{H}_{\sqrt{N}}$, for counting the running times of loops. $\tilde{H}$ is a Hadamard-like unitary to create uniform superposition on q_x, q_y , which is composited by local operations that only allow transition on adjacent position, e.g., $|x, y\rangle$ and $|x-1 \bmod \sqrt{N}, y\rangle$. C is the marking coin operator in Reference [3] and $S(\theta_1, \theta_2) = e^{-i\theta_2|+\rangle_{c_y}\langle+|} e^{-i\theta_1|+\rangle_{c_x}\langle+|} S_m$ is the parameterized shift operator, which can be implemented by a subprogram as follows:

$$c_x, c_y, q_x, q_y := S_m[c_x, c_y, q_x, q_y]; c_x := e^{-i\theta_1|+\rangle_{c_x}\langle+|}[c_x]; c_y := e^{-i\theta_2|+\rangle_{c_y}\langle+|}[c_y].$$

In particular, we have $S(0, 0) = S_m$ and $S(\pi, \pi) = S_f$. $A = \sum_{n=0}^{\sqrt{N}-1} |n+1\rangle\langle n| + |0\rangle\langle\sqrt{N}|$ adds t by 1 in every loop and measurement M' checks the value of t by $M' = \{M'_0 = \sum_{n=0}^{\sqrt{N}-1} |n\rangle_t\langle n|, M'_1 = I - M'_0\}$.

In this experiment, we choose $N = 16$, and the grid is $\{(i, j) : 0 \leq i, j \leq 3\}$. The loop measurement M is $\{M_0 = |3\rangle_{q_x}\langle 3| \otimes |3\rangle_{q_y}\langle 3|, M_1 = I - M_0\}$ with $(3, 3)$ being the marked vertex for convenience. The input state can be arbitrary, since all variables in $P_2(\theta_1, \theta_2)$ will be initialized. The observable we choose is $O_2 = \frac{1}{\sqrt{N}} \sum_{n=1}^{\sqrt{N}} n|n\rangle_t\langle n|$, which expresses the running number of loop

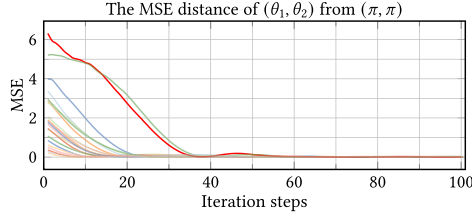


Fig. 8. MSE distance with respect to the iteration steps in optimizing $P_2(\theta_1, \theta_2)$. Differently colored lines represent 21 experiments with randomly initialized parameters (θ_1, θ_2) .

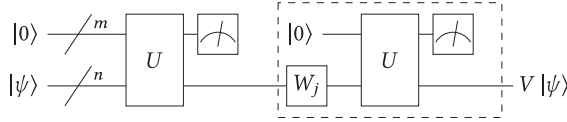


Fig. 9. RUS design circuit to implement unitary V [14].

iterations as the O_1 in Section 7.1. Our **target** is to *identify shift operator $S(\theta_1^*, \theta_2^*)$ that minimizes the expectation function of observable O_2 with $P_2(\theta_1, \theta_2)$.*

We summarize below the needed configuration in the experiment workflow.

Given: Parameterized QW program $P_2(\theta_1, \theta_2)$, arbitrary input state ρ , and observable O_2 .

Workflow: In (2), samples' number: 2×10^4 empirically chosen with details in Appendix C. In (3), Adam's setting: $\beta_1 = 0.9, \beta_2 = 0.999, \alpha = 0.1$; initial parameter: θ_1, θ_2 are randomly initialized in $2\pi \times [0.1, 0.9]$ to avoid certain extreme cases when (θ_1, θ_2) is close to $(0, 0)$;

Goal: **minimize** the expectation function of observable O_2 .

Results. During the optimizing process, we recorded the **Mean-Squared-Error (MSE)** distance⁸ of parameters (θ_1, θ_2) from (π, π) , which is shown in Figure 8. Each colored line in Figure 8 represents an independent optimization with different initial parameters. In particular, the initial parameters of the red line in Figure 8 are manually set to $(0.2\pi, 0.2\pi)$ to be far away from (π, π) . All independent training optimizing threads converge to the shift operator $S(\pi, \pi) = S_f$ after 60 steps, which recovers the operator S_f by human design [3], automatically in our experiment.

7.3 Repeat-until-success Unitary Implementation

In this subsection, we demonstrate that our framework can learn realizable instances of **repeat-until-success (RUS)** circuits. RUS depicts a design pattern, repeating an operation until getting the desired result, which has been widely used in quantum circuit design [14, 43, 53, 67]. A general layout of RUS circuits [14] is shown in Figure 9, where the dashed part is always applied if the measurement outcome is undesirable. Notice that W_j in Figure 9 is designed to restore the state of the system to $|0\rangle|\psi\rangle$ based on the measurement outcome, as only one copy of $|\psi\rangle$ is provided. The RUS circuits have been shown to achieve a better (expected) depth over ancilla-free techniques for single-qubit unitary decomposition [14, 53].

Parameterized RUS Program. Consider the program:

$$\begin{aligned} P_3(\theta_1, \theta_2, \theta_3) \equiv & r := |0\rangle; q, r := U[q, r]; \\ & \textbf{while } M[r] = 1 \textbf{ do } q := W(\theta_1, \theta_2, \theta_3)[q]; r := |0\rangle; q, r := U[q, r]; \textbf{ od}, \end{aligned} \quad (7.2)$$

⁸MSE distance between (θ_1, θ_2) and (π, π) is $\frac{1}{2}((\theta_1 - \pi)^2 + (\theta_2 - \pi)^2)$.

where q is a qubit variable and r is an ancilla qubit variable, measurement $M = \{M_0 = |0\rangle_r\langle 0|, M_1 = |1\rangle_r\langle 1|\}$. We are provided with a unitary U that will induce the desired operation on q if the outcome of performing measurement M after execution of $r := |0\rangle; q, r := U[q, r]$ is 0, otherwise, we need a recovery operation $W(\theta_1, \theta_2, \theta_3) = e^{-i\theta_1|0\rangle_q\langle 0|}e^{-i\theta_2|+\rangle_q\langle +|}e^{-i\theta_3|0\rangle_q\langle 0|}$, a fully parameterized single-qubit unitary (Z-X decomposition in Reference [52]) that can be implemented by a subprogram as follows,

$$q := e^{-i\theta_3|0\rangle_q\langle 0|}[q]; q := e^{-i\theta_2|+\rangle_q\langle +|}[q]; q := e^{-i\theta_1|0\rangle_q\langle 0|}[q],$$

to restore the state of q and repeat the whole process until obtaining the outcome 0.

In this experiment, the unitary U in program $P_3(\theta_1, \theta_2, \theta_3)$ is chosen as $(|0\rangle_r\langle 0| \otimes V_1 + |1\rangle_r\langle 1| \otimes V_2)(H \otimes I)$ with randomly generated single-qubit unitaries V_1 and V_2 . Our **target** is to *identify suitable parameters $\theta_1^*, \theta_2^*, \theta_3^*$ such that program $P_3(\theta_1^*, \theta_2^*, \theta_3^*)$ acts as the same as the unitary V_1 for an information-complete basis*

$$\{|\psi_1\rangle = |0\rangle, |\psi_2\rangle = |1\rangle, |\psi_3\rangle = |+\rangle = (|0\rangle + |1\rangle)/\sqrt{2}, |\psi_4\rangle = |Y_+\rangle = (|0\rangle + i|1\rangle)/\sqrt{2}\}$$

of variable q . That is, for any $1 \leq j \leq 4$, $\llbracket P_3(\theta_1^*, \theta_2^*, \theta_3^*) \rrbracket (|\psi_j\rangle_q\langle \psi_j|) = V_1|\psi_j\rangle_q\langle \psi_j|V_1^\dagger$, which is equivalent to $\text{tr}(V_1|\psi_j\rangle_q\langle \psi_j|V_1^\dagger \llbracket P_3(\theta_1^*, \theta_2^*, \theta_3^*) \rrbracket (|\psi_j\rangle_q\langle \psi_j|)) = 1$. Therefore, we choose four pairs of input states and observable ($\rho_j = |\psi_j\rangle_q\langle \psi_j|, O_{3,j} = V_1|\psi_j\rangle_q\langle \psi_j|V_1^\dagger, 1 \leq j \leq 4$ and denote the expectation function of program $P_3(\theta_1, \theta_2, \theta_3)$ with respect to input state ρ_j and observable $O_{3,j}$ as $f_j(\theta_1, \theta_2, \theta_3)$ for $1 \leq j \leq 4$. To optimize the four functions $f_j, 1 \leq j \leq 4$ simultaneously close to 1, we introduce a MSE loss function $l(\theta_1, \theta_2, \theta_3) = \frac{1}{4} \sum_{j=1}^4 (f_j(\theta_1, \theta_2, \theta_3) - 1)^2$ to be minimized in the experiment workflow.

We summarize below the needed configuration in the experiment workflow.

Given: Parameterized RUS program $P_3(\theta_1, \theta_2, \theta_3)$ and four pairs of input state and observable $(\rho_j, O_{3,j}), 1 \leq j \leq 4$.

Workflow: In (2), samples' number: 4.7×10^4 empirically chosen with details in Appendix C. In (3), Adam's setting: $\beta_1 = 0.9, \beta_2 = 0.999, \alpha = 0.2$; initial parameter: θ_1, θ_2 are randomly initialized; goal: **minimize** the MSE loss function $l(\theta_1, \theta_2, \theta_3)$.

Results. We did 10 independent optimizations. In each optimization, V_1 and V_2 are randomly generated. With the iteration steps less than 60, the MSE loss l can be reduced to 0.0001, which implies f_j is greater than 0.98 for all j . Because $\{|\psi_j\rangle_q\langle \psi_j|\}_{j=1}^4$ forms a complete basis of $\mathcal{D}(\mathcal{H}_q)$, we can conclude that the program P_3 produces an approximate operation of V_1 that we want in each optimization. Therefore, the experimental result confirms our framework's feasibility and validates the experiment workflow for automatically getting suitable parameters.

8 CONCLUSION

In this article, we have studied the AD of quantum programs with unbounded loops. We find a sufficient condition—finite-dimensional state spaces—for quantum programs' differentiability. This sufficient condition is reasonable and terse in practical applications. Under this condition, we build a source-level code transformation with correctness proof to achieve AD for quantum programs. For the effectiveness of our approach, we give a result of sample complexity that is comparable to previous work of bounded loops. We also implement our AD and demonstrate the feasibility of our AD by three examples: parameterized amplitude amplification, quantum walk-based search algorithm, and repeat-until-success unitary implementation.

Our research enables the automatic optimization of complex quantum programs without requiring manual derivation. We hope that it will provide a deeper understanding of differentiable quantum programming, provide a theoretical basis for the development of quantum machine learning

software frameworks, and expect to use it to discover new quantum algorithms, especially with unbounded loops.

APPENDICES

A PRACTICAL VARIANCE BOUND FOR DIFFERENTIAL PROGRAMS

In this section, we give fine bounds of $\langle O_d^2 \otimes O^2 \rangle$ and $\langle O_d^4 \otimes O^4 \rangle$ that are used in our case studies for estimating the number of samples. Before that, we give the formal definitions of the two previous notions, $RC_\theta(P(\theta))$ and $LC(P(\theta))$.

Definition A.1. The “Running Count for θ ” in $P(\theta)$, denoted $RC_\theta(P(\theta))$, is defined inductively on the program structure:

- $RC_\theta(P(\theta)) = 0$ for $P(\theta) \equiv \text{skip}$, $q := |0\rangle$ or $\bar{q} := U[\bar{q}]$.
- If $P(\theta) \equiv \bar{q} := e^{-i\theta'\sigma}[\bar{q}]$, then $RC_\theta(P(\theta)) = 1$ when θ' is θ ; otherwise, $RC_\theta(P(\theta)) = 0$.
- If $P(\theta) \equiv P_1(\theta); P_2(\theta)$, then $RC_\theta(P(\theta)) = RC_\theta(P_1(\theta)) + RC_\theta(P_2(\theta))$.
- If $P(\theta) \equiv \text{if } (\Box m \cdot M[\bar{q}] = m \rightarrow P_m(\theta)) \text{ fi}$, then $RC_\theta(P(\theta)) = \max_m RC_\theta(P_m(\theta))$.
- If $P(\theta) \equiv \text{while } M[\bar{q}] = 1 \text{ do } Q(\theta) \text{ od}$, then $RC_\theta(P(\theta)) = RC_\theta(Q(\theta))$.

Definition A.2. The “Loop Count” in $P(\theta)$, denoted $LC(P(\theta))$, is defined by induction on the program structure as follows:

- $LC(P(\theta)) = 0$ for $P(\theta) \equiv \text{skip}$, $q := |0\rangle$, $\bar{q} := U[\bar{q}]$ or $\bar{q} := e^{-i\theta'\sigma}[\bar{q}]$.
- If $P(\theta) \equiv P_1(\theta); P_2(\theta)$, then $LC(P(\theta)) = LC(P_1(\theta)) + LC(P_2(\theta))$.
- If $P(\theta) \equiv \text{if } (\Box m \cdot M[\bar{q}] = m \rightarrow P_m(\theta)) \text{ fi}$, then $LC(P(\theta)) = \sum_m LC(P_m(\theta))$.
- If $P(\theta) \equiv \text{while } M[\bar{q}] = 1 \text{ do } Q(\theta) \text{ od}$, then $LC(P(\theta)) = LC(Q(\theta)) + 1$.

THEOREM A.3. *In the same setting as in Theorem 5.4, for a fixed θ , if all the **while**-statements (subprograms) in $P(\theta)$ terminate almost surely, then the expectation of $O_d^2 \otimes O^2$:*

$$\langle O_d^2 \otimes O^2 \rangle = \text{tr} \left(O_d^2 \otimes O^2 \left\| \frac{\partial}{\partial \theta}(P(\theta)) \right\| (\rho) \right)$$

is upper-bounded by

$$M^2 \left(4S(M_1) + \sum_{k=1}^{\infty} \left((M_2 + (k-1)(k^{M_2-1} - 1)_+) S((k+1)^{M_2} M_1 - 1) \left(2\epsilon^{\lfloor \frac{k-1}{N_\epsilon} \rfloor} + 2\epsilon^{\lfloor \frac{k-1}{N_\epsilon} \rfloor - 1} \right) \right) \right), \quad (\text{A.1})$$

where

- M is the largest eigenvalue of $|O|$;
- $M_1 = RC_\theta(P(\theta))$, $M_2 = LC(P(\theta))$;
- μ is the distribution we adopted in code transformation rules and satisfies *converging-rate condition*;
- $S(n) \equiv \sum_{j=1}^n 1/\mu(j)$ for every $n \geq 1$; $(x)_+ \equiv \max\{0, x\}$;
- $\epsilon \in (0, 1)$ and N_ϵ is the largest number of N in Lemma 5.3 that is applied to all M_2 loop statements in $P(\theta)$.

PROOF. See Appendix D.5. □

The *converging-rate condition* ensures

$$\lim_{n \rightarrow \infty} \sqrt[n]{S(n)} = 1.$$

Thus, the infinite summation terms in Equation (A.1) have exponential damping factor $\epsilon^{\frac{1}{N_e}} < 1$, then the summation is convergent. To give a clear sense of this bound, we can derive a corollary when we use the distribution mentioned in Equation (6.1).

COROLLARY A.4. *In the same setting as in Theorem A.3, let the distribution μ be*

$$\mu(j) = \frac{1}{c(s)j \ln^{1+s}(j+e)},$$

where $c(s) = \sum_{j=1}^{\infty} 1/(j) / \ln^{1+s}(j+e) < \infty, s \in (0, 1]$. We have the expectation of $O_d^2 \otimes O^2$ is upper-bounded by

$$M^2 \left(4T(M_1) + \sum_{k=1}^{\infty} \left((M_2 + (k-1)(k^{M_2-1} - 1)_+) T \left((k+1)^{M_2} M_1 - 1 \right) \left(2\epsilon^{\lfloor \frac{k-1}{N_e} \rfloor} + 2\epsilon^{\lfloor \frac{k-1}{N_e} \rfloor - 1} \right) \right) \right), \quad (\text{A.2})$$

where $T(n) = \frac{c(s)}{2} x^2 \ln^{1+s}(x+e)$ is an upper bound of $S(n)$ by integral.

With $T(n)$ substituted, Corollary A.4 implies

$$\langle O_d^2 \otimes O^2 \rangle \leq 2M^2 c(s) \left(M_1^2 \ln^{1+s}(M_1+e) + M_1^{3+s} M_2^{2+s} \sum_{k=1}^{\infty} (k+1)^{3M_2+1} \epsilon^{\frac{k-1}{N_e} - 2} \right).$$

The infinite summation $\sum_{k=1}^{\infty} (k+1)^{3M_2+1} \epsilon^{\frac{k-1}{N_e} - 2}$ is related to Eulerian polynomials [6] and can be easily bounded by

$$\frac{(3M_2+1)!}{\epsilon^{2+\frac{1}{N_e}} (1 - \epsilon^{\frac{1}{N_e}})^{3M_2+2}}.$$

Thus, we get the bound for $\langle O_d^2 \otimes O^2 \rangle$ that implies Theorem 6.2:

$$2M^2 c(s) \left(M_1^2 \ln^{1+s}(M_1+e) + \frac{M_1^{3+s} M_2^{2+s} ((3M_2+1)!)}{\epsilon^{2+\frac{1}{N_e}} (1 - \epsilon^{\frac{1}{N_e}})^{3M_2+2}} \right).$$

For bound of $\langle O_d^4 \otimes O^4 \rangle$, we have a theorem similar to Theorem A.3.

THEOREM A.5. *In the same setting as in Theorem 5.4, for a fixed θ , if all the **while**-statements (subprograms) in $P(\theta)$ terminate almost surely, then the expectation of $O_d^4 \otimes O^4$:*

$$\langle O_d^4 \otimes O^4 \rangle = \text{tr} \left(O_d^4 \otimes O^4 \left\| \frac{\partial}{\partial \theta} (P(\theta)) \right\| (\rho) \right)$$

is upper-bounded by

$$4M^2 \left(4S'(M_1) + \sum_{k=1}^{\infty} \left((M_2 + (k-1)(k^{M_2-1} - 1)_+) S' \left((k+1)^{M_2} M_1 - 1 \right) \left(2\epsilon^{\lfloor \frac{k-1}{N_e} \rfloor} + 2\epsilon^{\lfloor \frac{k-1}{N_e} \rfloor - 1} \right) \right) \right), \quad (\text{A.3})$$

where

- M is the largest eigenvalue of $|O|$;
- $M_1 = RC_{\theta}(P(\theta))$, $M_2 = LC(P(\theta))$;
- μ is the distribution we adopted in code transformation rules and satisfies *converging-rate condition*;
- $S'(n) \equiv \sum_{j=1}^n 1/\mu^3(j)$ for every $n \geq 1$; $(x)_+ \equiv \max\{0, x\}$;

$-\epsilon \in (0, 1)$ and N_ϵ is the largest number of N in Lemma 5.3 that is applied to all M_2 loop statements in $P(\theta)$.

PROOF. The Proof is similar to Theorem A.3. □

When unbounded loops exist (i.e., $M_2 > 0$), the bound given in Theorem A.3 will depend on the number N of iterations in Lemma 5.3. We provide the following variant of Lemma 5.3 for a practically more approachable calculation, which, in particular, has been used in determining the number of samples in our case studies:

LEMMA A.6. Consider a quantum loop $P \equiv \text{while } M[\bar{q}] = 1 \text{ do } Q \text{ od}$ with fixed parameters (omitted) and a finite dimension of $\mathcal{H}_P$; we define superoperators $\mathcal{E}_i : \mathcal{D}(\mathcal{H}_P) \rightarrow \mathcal{D}(\mathcal{H}_P)$ by $\mathcal{E}_i(\rho) = M_i \rho M_i^\dagger$, $i = 0, 1$ and $\mathcal{E} : \mathcal{D}(\mathcal{H}_P) \rightarrow \mathcal{D}(\mathcal{H}_P)$ by $\mathcal{E}(\rho) = \llbracket Q \rrbracket(\rho)$. Assume that the Kraus representation of $\mathcal{E} \circ \mathcal{E}_1$ is $\sum_j E_j(\cdot) E_j^\dagger$. We write $E = \sum_j E_j \otimes E_j^*$. If E can be diagonalized, and there exists $\epsilon < 1$ such that the module of E 's eigenvalues is either equal to 1 or less than ϵ , then for $\forall n \in \mathbb{N}, \forall \rho \in \mathcal{D}(\mathcal{H}_P)$,

$$\text{tr}(\mathcal{E}_0 \circ (\mathcal{E} \circ \mathcal{E}_1)^n(\rho)) \leq \epsilon^n \text{tr}(\rho).$$

PROOF. This lemma can be proved using the techniques developed in Reference [75, Section 5], where it was proved that the module of all E 's eigenvalues is less than or equal to 1. We omit the details here. □

B AUTOMATIC DIFFERENTIATION BASED ON PARAMETER-SHIFT RULE

Parameter-shift Rule. A way for differentiation of Hamiltonian simulation $e^{-i\theta H}$ with H having at most two distinct eigenvalues, called the parameter-shift rule, was given in References [49, 59]. For those Hamiltonians with more than two distinct eigenvalues, the differentiation can be obtained via **LCU (Linear Combination of Unitaries)** [18]. We also note some recent independent developments [35, 39, 68] of variants of the parameter-shift rules to handle more general $e^{-i\theta H}$. But here, for the sake of convenience, we still only use the previous parameter-shift rule, which is adopted in Zhu et al. [77]'s work on differentiable quantum programming.

Let us consider a simple example: the expectation function $f(\theta) = \text{tr}(O e^{-i\theta X} \rho e^{i\theta X})$. We can check that

$$\frac{d}{d\theta} f(\theta) = f\left(\theta + \frac{\pi}{4}\right) - f\left(\theta - \frac{\pi}{4}\right).$$

More generally, if the Hamiltonian H has only two eigenvalues $\pm r$, $r > 0$ and

$$f(\theta) = \text{tr}(O e^{-i\theta H} \rho e^{i\theta H}),$$

then

$$\frac{d}{d\theta} f(\theta) = r \left(f\left(\theta + \frac{\pi}{4r}\right) - f\left(\theta - \frac{\pi}{4r}\right) \right).$$

Although this form looks like a finite difference, it does express the exact derivative of f rather than an approximate value. Therefore, the derivative can be obtained by shifting a single gate parameter. It is worth mentioning that the same differentiation was effectively achieved in Reference [77] using one extra ancilla as the control qubit to create a superposition of two quantum circuits.

The parameter-shift rule can be used as an alternative to the commutator form rule in the DSOP part of Figure 3. Furthermore, we can construct code transformation rules for AD

$$\begin{aligned}
T_\theta(\text{skip}) &\equiv \text{skip} \\
T_\theta(q := |0\rangle) &\equiv q := |0\rangle \\
T_\theta(\bar{q} := U[\bar{q}]) &\equiv \bar{q} := U[\bar{q}] \\
T_\theta(\bar{q} := U(\theta')[\bar{q}]) &\equiv \bar{q} := U(\theta')[\bar{q}] \quad (\theta' \neq \theta) \\
T_\theta(P_1(\theta); P_2(\theta)) &\equiv T_\theta(P_1(\theta)); T_\theta(P_2(\theta)) \\
T_\theta(\text{if } (\Box m \cdot M[\bar{q}] = m \rightarrow P_m(\theta)) \text{ fi}) &\equiv \text{if } (\Box m \cdot M[\bar{q}] = m \rightarrow T_\theta(P_m(\theta))) \text{ fi} \\
T_\theta(\text{while } M[\bar{q}] = 1 \text{ do } P(\theta) \text{ od}) &\equiv \text{while } M[\bar{q}] = 1 \text{ do } T_\theta(P(\theta)) \text{ od} \\
(*) \quad T_\theta(\bar{q} := U(\theta)[\bar{q}]) &\equiv \text{if } (M_{q_1, q_2}[q_1, q_2] = 0 \rightarrow q_c := C[q_c]; q_c, q_2 := GP[q_c, q_2] \\
&\quad \Box = 1 \rightarrow q_1 := X[q_1]; A := H[A]; \\
&\quad \quad A, \bar{q} := C_-U(\theta)[A, \bar{q}]; A := H[A] \\
&\quad \Box = 2 \rightarrow \text{skip}) \text{ fi}; \bar{q} := e^{-i\theta\sigma}[\bar{q}].
\end{aligned}$$

Fig. 10. Code transformation rules for T_θ , where $M_{q_1, q_2} = \{M_0 = |00\rangle\langle 00|, M_1 = |01\rangle\langle 01|, M_2 = |10\rangle\langle 10| + |11\rangle\langle 11|\}$, $C = \sum_{j=0}^{\infty} |j+1\rangle\langle j|$ is the right-translation operator, $GP = \sum_{j=1}^{\infty} |j\rangle\langle j| \otimes R_y(2 \arcsin(\sqrt{b_j}))$ and $b_j = \mu(j)/(1 - \sum_{k=1}^{j-1} \mu(k))$.

based on the parameter-shift rule, as we did based on the commutator form rule in the main body.

AD by Code Transformations. We only considered the parameterized forms used by Reference [77]. That is, our parameterized unitary $U(\theta)$ is chosen from Pauli rotations

$$\left\{ R_\sigma(\theta) = e^{-i\frac{\theta}{2}\sigma}, R_{\sigma \otimes \sigma}(\theta) = e^{-i\frac{\theta}{2}\sigma \otimes \sigma} : \sigma = X, Y, Z; \theta \in \Theta \right\}. \quad (\text{B.1})$$

We construct a code transformation operation T_θ in Figure 10, where

$$C_-U(\theta) = |0\rangle_A\langle 0| \otimes U(\theta) + |1\rangle_A\langle 1| \otimes U(\pi)$$

(see Reference [77] for more detail of the definition of C_-U), with $U(\theta)$ being in the form of Pauli rotations and A an ancilla quantum variable. Then, we have the following theorem:

THEOREM B.1. *Given a quantum program $P(\theta)$ that is parameterized by Pauli rotations in Equation (B.1), an initial state ρ , an observable O on $\mathcal{H}_{P(\theta)}$ of a finite dimension. Let*

$$\frac{\partial}{\partial \theta}(P(\theta)) \equiv q_1 := |0\rangle; q_2 := |0\rangle; q_c := |0\rangle; A := |0\rangle; q_c := C[q_c]; q_c, q_2 := GP[q_c, q_2]; T_\theta(P(\theta)).$$

Then,

$$\frac{\partial}{\partial \theta}(\text{tr}(O[P(\theta)](\rho))) = \text{tr}\left(Z_A \otimes O_c \otimes O \left[\frac{\partial}{\partial \theta}(P(\theta)) \right] (\rho)\right),$$

where $Z_A = |0\rangle_A\langle 0| - |1\rangle_A\langle 1|$, and

$$O_c = \sum_{j=1}^{\infty} \frac{1}{\mu(j)} |j\rangle\langle j| \otimes |1\rangle\langle 1|$$

is an observable on $\mathcal{H}_{q_c} \otimes \mathcal{H}_{q_1}$.

PROOF. The proof is similar to that of Theorem 5.4. The only difference is that in this proof, for every computation path π and the subset A_π we mentioned in the proof of Theorem 5.4, let $\mathcal{E}_\eta$

denote the superoperator of η for any path η ; then, we need the following result:

$$\frac{\partial}{\partial \theta}(\text{tr}(O\mathcal{E}_\pi(\rho))) = \text{tr}\left(Z_A \otimes O_c \otimes O \sum_{\eta \in A_\pi} \mathcal{E}_\eta(\rho)\right),$$

which is guaranteed by the soundness theorem in Reference [77] (Theorem 6.2 therein). $\square$

Given that all parameterized quantum programs with bounded loops considered in Reference [77] are defined in the setting of Pauli rotations, the above theorem (together with code transformation T_θ) strictly improves the corresponding result of Reference [77] with unbound loops.

C NUMBER OF SAMPLES IN CASE STUDIES

In this section, we elaborate on how to determine the number of samples for estimating the expectation function of differential programs in case studies. Our analysis mainly relies on the bound of $\langle O_d^2 \otimes O^2 \rangle$ and $\langle O_d^4 \otimes O^4 \rangle$ in Appendix A. Although the variance bound we proved in Theorem A.3 matches the one of Zhu et al. [77] when there is no unbounded loop, it is still not scalable in practical applications. In the following analysis of case studies, the actual value of $\langle O_d^2 \otimes O^2 \rangle$ is much less than the bound we prove.

C.1 Parameterized Amplitude Amplification

It is a bit troublesome to directly estimate the variance bound of $P_1(\theta)$. We need an auxiliary program:

```

 $Q(\theta) \equiv q := |0\rangle; r := |0\rangle; q := A[q];$ 
  while  $M[r] = 1$  do
     $q := Z[q]; q := A^\dagger[q]; q := Z[q]; q := A[q];$ 
     $q, r := e^{-i\theta\sigma_{[1]}(1 \otimes Y)}[q, r]$ 
  od

```

This $Q(\theta)$ does not contain the quantum variable t in the program $P_1(\theta)$, but its behavior is similar to $P_1(\theta)$. Its differential program $\frac{\partial}{\partial \theta}(Q(\theta))$ is also similar to $\frac{\partial}{\partial \theta}(P_1(\theta))$. Consider the observable $\tilde{O}_1 = I$ for program $Q(\theta)$, we can conclude that the expectation $\langle O_d^2 \otimes \tilde{O}_1^2 \rangle$ of $\frac{\partial}{\partial \theta}(P_1(\theta))$ is less than the expectation $\langle O_d^2 \otimes \tilde{O}_1^2 \rangle$ of $\frac{\partial}{\partial \theta}(Q(\theta))$:

- (1) Observable O_1 for $P_1(\theta)$ yields the result that is equal to or less than 1 and observable $\tilde{O}_1$ for $Q(\theta)$ leads to the result 1, which indicates that the output of $Q(\theta)$ is always greater than $P_1(\theta)$.
- (2) Since the differential program keeps the same structure as the original program, the above result also holds for program $\frac{\partial}{\partial \theta}(Q(\theta))$ with observable $O_d^2 \otimes \tilde{O}_1^2$ and program $\frac{\partial}{\partial \theta}(P_1(\theta))$ with observable $O_d^2 \otimes O_1^2$ if they have executed the same branches.

Therefore, the expectation $\langle O_d^2 \otimes O_1^2 \rangle$ of $\frac{\partial}{\partial \theta}(P_1(\theta))$ is less than the expectation $\langle O_d^2 \otimes \tilde{O}_1^2 \rangle$ of $\frac{\partial}{\partial \theta}(Q(\theta))$.

The Theorem A.3 with the fact that $Q(\theta)$ meets the conditions of Lemma A.6 can give us an upper bound of $\langle O_d^2 \otimes \tilde{O}_1^2 \rangle$. When $p = 1/100$ and $\theta = 4 \arccos((1 - 2\sqrt{p(1-p)})/(1 + 2\sqrt{p(1-p)})) = 3.3568$, we numerically calculate the ϵ for $Q(\theta)$ in Lemma A.6 as 0.6681. With $M = 1, M_1 = 1, M_2 = 1$ in Theorem A.3, we obtain 799.72 as an upper bound of $\langle O_d^2 \otimes \tilde{O}_1^2 \rangle$. However, the bound for $\langle O_d^4 \otimes \tilde{O}_1^4 \rangle$ in Theorem A.5 can also be numerically calculated as 1.013×10^7 . By Chebyshev's Inequality, we use $1.013 \times 10^7 / 30^2 / 0.1 \approx 1 \times 10^5$ samples to sample $\langle O_d^2 \otimes O_1^2 \rangle \approx 14.26$ in an error of 30 with

failure probability less than 10%. Thus, we can use $30 + 14.26 = 44.26$ as the actual value of $\langle O_d^2 \otimes O_1^2 \rangle$. This 44.26 is much less than 799.72. By Chebyshev's Inequality, to estimate $\langle O_d \otimes O_1 \rangle$ in precision $\delta = 0.1$ with failure probability less than $c = 10\%$, the number of samples we need is less than $\text{Var}(O_d \otimes O_1)/(\delta^2 c) \leq 4.426 \times 10^4$. In this experiment, we use $5/\sqrt{p} \times 10^3 (= 5 \times 10^4$ when $p = 1/100)$ samples for each p .

C.2 Quantum Walk with Parameterized Shift Operator

To estimate the number of samples for $P_2(\theta_1, \theta_2)$, we need another similar program as follows:

```

 $Q(\theta_1, \theta_2) \equiv t := |0\rangle; \text{while } M[q_x, q_y] = 1 \text{ do}$ 
     $c_x := |0\rangle; c_y := |0\rangle; q_x := |0\rangle; q_y := |0\rangle;$ 
     $c_x := H[c_x]; c_y := H[c_y]; q_x, q_y := \tilde{H}[q_x, q_y];$ 
     $c_x, c_y, q_x, q_y := C[c_x, c_y, q_x, q_y]; c_x, c_y, q_x, q_y := S(\theta_1, \theta_2)[c_x, c_y, q_x, q_y];$ 
     $c_x, c_y, q_x, q_y := C[c_x, c_y, q_x, q_y]; c_x, c_y, q_x, q_y := S_m[c_x, c_y, q_x, q_y];$ 
    if  $(M'[t] = 0 \rightarrow A[t] \square = 1 \rightarrow \text{skip})$  fi od.

```

The second shift operator in $Q(\theta_1, \theta_2)$ is not parameterized. But its behavior is the same as $P(\theta_1, \theta_2)$. Thus, we only need to estimate the number of samples for $Q(\theta_1, \theta_2)$. When $(\theta_1, \theta_2) = (\pi, \pi)$, we can numerically calculate the ϵ for $Q(\theta_1, \theta_2)$ in Lemma A.6 is less than 0.76. With $M = 1, M_1 = 1, M_2 = 1$ in Theorem A.3, we obtain 2,230.86 as an upper bound of variance. The bound for $\langle O_d^4 \otimes O^4 \rangle$ in Theorem A.5 is 8.14×10^7 . Then, we use $8.14 \times 10^7 / 20^2 / 0.1 \approx 2 \times 10^5$ samples to sample $\langle O_d^2 \otimes O^2 \rangle \approx 104.23$ in an error of 20 with failure probability less than 10%. Thus, we can use $104.23 + 20 = 124.23$ as the actual value of $\langle O_d^2 \otimes O^2 \rangle$. By Chebyshev's Inequality, to estimate $\langle O_d \otimes O_2 \rangle$ in precision $\delta = 0.1$ with failure probability less than $c = 10\%$, the number of samples we need is less than 1.24×10^5 .

However, this number of samples is large for us, as the simulation of $P_2(\theta_1, \theta_2)$ takes a lot of time in Q#. In this experiment, we choose 2×10^4 as the number of samples. Our experiment shows that this number of samples is already good for training. This phenomenon has been studied in the optimization of PQCs (VQCs): Sweke et al. [63] found that even using single measurement outcomes for estimation of expectation values is sufficient in optimization algorithms, which results in a form of *stochastic* gradient descent optimization [57].

C.3 Repeat-until-success Unitary Implementation

It is easy to see that $P_3(\theta_1, \theta_2, \theta_3)$ satisfies the conditions of Lemma A.6 with $\epsilon = 0.5$. With $M = 1, M_1 = 1, M_2 = 1$ for each parameter, the variance bound in Theorem A.3 is 243.19. While the estimated value of $\langle O_d^2 \otimes O^2 \rangle$ is 15.298.

The partial derivative of $l(\theta_1, \theta_2, \theta_3)$ with respect to θ_1 is

$$\frac{\partial l}{\partial \theta_1} = \frac{1}{2} \sum_{j=1}^4 (E_j - 1) \frac{\partial E_j}{\partial \theta_1}.$$

Suppose E_j and $\partial E_j / \partial \theta_1$ are estimated in precision δ_1 and δ_2 , respectively. Then, $\partial l / \partial \theta_1$ is in precision

$$\begin{aligned} \frac{1}{2} \sum_{j=1}^4 \left(|E_j - 1| \delta_2 + \left| \frac{\partial E_j}{\partial \theta_1} \right| \delta_1 + \delta_1 \delta_2 \right) &= \left(\frac{1}{2} \sum_{j=1}^4 |E_j - 1| \right) \delta_2 + \left(\frac{1}{2} \sum_{j=1}^4 \left| \frac{\partial E_j}{\partial \theta_1} \right| \right) \delta_1 + 2\delta_1 \delta_2 \\ &\equiv A\delta_1 + B\delta_2 + 2\delta_1 \delta_2. \end{aligned}$$

To limit it in 0.1, we can choose $\delta_1 \sim 0.01/B$, $\delta_2 \sim 0.09/A$. Then, assume that during most of the training process, $A \leq \frac{1}{2} \times 4 \times 0.3 = 0.6$, we have $\delta_2 \leq 0.15$. By Chebyshev's Inequality, to estimate $\partial E_j / \partial \theta_1$ in precision $\delta = 0.15$ with failure probability less than $c = 1\%$, the number of samples we need is less than 4.72×10^4 . With this number of samples, the probability of all $\partial E_j / \partial \theta_1, j = 1, 2, 3, 4$ are estimated in 0.15 is greater than $0.99^4 = 92.2\%$.

D DETAILED PROOFS

D.1 Proof of Lemma 5.3

Before giving proof details of Lemma 5.3, we need some lemmas and definitions. For those who want a more detailed understanding of this subsection, you can refer to quantum graph theory and quantum Markov chains [30, 73, 76].

With the same notations of Lemma 5.3, we first list other needed notations:

- $\mathcal{G} = \mathcal{E} \circ \mathcal{E}_1$.
- $\sigma = \lim_{n \rightarrow \infty} \frac{1}{n} \sum_{k=0}^n \mathcal{G}^n(I_{\mathcal{H}_P})$, where $I_{\mathcal{H}_P}$ is the identify operator on $\mathcal{H}_P$. The existence of σ is easily obtained; moreover, $\lim_{n \rightarrow \infty} \frac{1}{n} \sum_{k=0}^n \mathcal{G}^n$ is also a superoperator [69].
- $\mathcal{Y} = \{|\psi\rangle \in \mathcal{H}_P \mid \langle\psi|\sigma|\psi\rangle = 0\}$, $\mathcal{X} = \text{supp}(\sigma) \equiv \mathcal{Y}^\perp$.
- P_X denotes the projector onto a space X , $I_{\mathcal{H}_P}$ denotes the identify operator on $\mathcal{H}_P$.
- The notation $|\psi\rangle \in \mathcal{H}$ for any Hilbert space $\mathcal{H}$ assumes $\| |\psi\rangle \| = \langle\psi|\psi\rangle = 1$ if there is no special remark.

LEMMA D.1 (MODIFIED FROM REFERENCE [76, THEOREM 1]).

- $\forall |\psi\rangle \neq 0 \in \mathcal{X}, \forall n \in \mathbb{N}. \text{tr}(P_X \mathcal{G}^n(|\psi\rangle\langle\psi|)) = \langle\psi|\psi\rangle = 1$.
- $\forall n \in \mathbb{N}. (\mathcal{G}^*)^n(P_X) \supseteq P_X$.

PROOF. Both of these propositions are trivial if $\mathcal{X}$ is Zero space. Thus, in the following, we assume $\mathcal{X}$ is not Zero space, which means $\sigma \neq 0$:

- According to Reference [52, P. 105], for any $|\psi\rangle \in \mathcal{X} = \text{supp}(\sigma)$, there exist $\lambda > 0$ and $\mu \in \mathcal{D}(\mathcal{X})$ such that $\sigma = \lambda|\psi\rangle\langle\psi| + \mu$, then

$$\begin{aligned} \text{tr}(\sigma) &= \text{tr}(P_X \sigma) = \text{tr}(P_X \mathcal{G}(\sigma)) = \dots = \text{tr}(P_X \mathcal{G}^n(\sigma)) \\ &= \lambda \text{tr}(P_X \mathcal{G}^n(|\psi\rangle\langle\psi|)) + \text{tr}(P_X \mathcal{G}^n(\mu)) \\ &\leq \lambda \text{tr}(|\psi\rangle\langle\psi|) + \text{tr}(\mu) = \text{tr}(\sigma). \end{aligned}$$

Because $\lambda > 0$, we conclude that

$$\text{tr}(P_X \mathcal{G}^n(|\psi\rangle\langle\psi|)) = \langle\psi|\psi\rangle = 1.$$

- The above statement tells that for any $|\psi\rangle \in \mathcal{X}$ and any $n \in \mathbb{N}$

$$1 = \text{tr}(P_X \mathcal{G}^n(|\psi\rangle\langle\psi|)) = \langle\psi|(\mathcal{G}^*)^n(P_X)|\psi\rangle$$

together with $I_{\mathcal{H}_P} \supseteq (\mathcal{G}^*)^n(I_{\mathcal{H}_P}) \supseteq (\mathcal{G}^*)^n(P_X)$, which means $\|(\mathcal{G}^*)^n(P_X)|\psi\rangle\| \leq 1$, we have

$$\forall |\psi\rangle \in \mathcal{X}, \forall n \in \mathbb{N}. |\psi\rangle = (\mathcal{G}^*)^n(P_X)|\psi\rangle.$$

Now, for any $|\alpha\rangle = x|\psi\rangle + y|\varphi\rangle \in \mathcal{H}_P$, where $|\psi\rangle \in \mathcal{X}, |\varphi\rangle \in \mathcal{Y}, |x|^2 + |y|^2 = 1$, we have

$$\begin{aligned} \langle\alpha|((\mathcal{G}^*)^n(P_X) - P_X)|\alpha\rangle &\geq \bar{x}y\langle\psi|(\mathcal{G}^*)^n(P_X)|\varphi\rangle + x\bar{y}\langle\varphi|(\mathcal{G}^*)^n(P_X)|\psi\rangle \\ &= \bar{x}y\langle\psi|\varphi\rangle + x\bar{y}\langle\varphi|\psi\rangle = 0. \end{aligned}$$

Thus, $(\mathcal{G}^*)^n(P_X) \supseteq P_X$ for all $n \in \mathbb{N}$.

□

LEMMA D.2.

— $\forall \epsilon \in (0, 1), \exists N > 0, \forall |\psi\rangle \in \mathcal{H}_P, \forall n > N,$

$$\text{tr}(P_Y \mathcal{G}^n(|\psi\rangle\langle\psi|)) < \epsilon.$$

— $\forall \epsilon \in (0, 1), \exists N > 0, (\mathcal{G}^*)^N(P_Y) \subseteq \epsilon P_Y.$

PROOF.

— For any $|\psi\rangle \in \mathcal{H}_P$, we first prove that

$$\lim_{n \rightarrow \infty} \text{tr}(P_Y \mathcal{G}^n(|\psi\rangle\langle\psi|)) = 0.$$

With $\mathcal{G}^*(P_X) \supseteq p_X$ in Lemma D.1 and $\mathcal{G}^*(P_X + P_Y) = \mathcal{G}^*(I_{\mathcal{H}_P}) \subseteq I_{\mathcal{H}_P} = P_X + P_Y$ by the definition of $\mathcal{G}$, we have $\mathcal{G}^*(P_Y) \subseteq P_Y$, which means $\text{tr}(P_Y \mathcal{G}^n(|\psi\rangle\langle\psi|)) \geq 0$ is non-increasing with $n \rightarrow \infty$, then there exists $a \geq 0$ such that

$$\lim_{n \rightarrow \infty} \text{tr}(P_Y \mathcal{G}^n(|\psi\rangle\langle\psi|)) = a.$$

Therefore,

$$\begin{aligned} 0 &= \text{tr}(P_Y \sigma) = \text{tr}\left(P_Y \lim_{n \rightarrow \infty} \frac{1}{n} \sum_{k=0}^n \mathcal{G}^k(I_{\mathcal{H}_P})\right) \\ &= \lim_{n \rightarrow \infty} \frac{1}{n} \sum_{k=0}^n \text{tr}(P_Y \mathcal{G}^k(I_{\mathcal{H}_P} - |\psi\rangle\langle\psi|)) + \lim_{n \rightarrow \infty} \frac{1}{n} \sum_{k=0}^n \text{tr}(P_Y \mathcal{G}^k(|\psi\rangle\langle\psi|)) \\ &\geq \lim_{n \rightarrow \infty} \frac{1}{n} \sum_{k=0}^n \text{tr}(P_Y \mathcal{G}^k(|\psi\rangle\langle\psi|)) \\ &= \lim_{n \rightarrow \infty} \text{tr}(P_Y \mathcal{G}^n(|\psi\rangle\langle\psi|)) \\ &= a \geq 0, \end{aligned}$$

which results $a = 0$. We define a series of continuous functions $f_n, n \in \mathbb{N}$ on $A = \{|\psi\rangle \mid |\psi\rangle \in \mathcal{H}_P\}$,

$$f_n(|\psi\rangle) = \text{tr}(P_Y \mathcal{G}^n(|\psi\rangle\langle\psi|)).$$

We have that f_n is monotonically decreasing and convergent to 0. Besides, $\mathcal{H}_P$ is finite-dimensional, then A is a compact set (unit sphere). By Dini's Theorem [58, Theorem 7.13], f_n is uniform convergent to 0, which is

$$\forall \epsilon > 0, \exists N > 0, \forall |\psi\rangle \in A, \forall n > N, |f_n(|\psi\rangle)| < \epsilon.$$

Therefore, $\forall \epsilon \in (0, 1), \exists N > 0, \forall |\psi\rangle \in \mathcal{H}_P, \forall n > N,$

$$\text{tr}(P_Y \mathcal{G}^n(|\psi\rangle\langle\psi|)) < \epsilon.$$

— According to the above, for any $\epsilon \in (0, 1)$, there exists $N_0 > 0$ such that $\forall |\varphi\rangle \in \mathcal{Y} \subseteq \mathcal{H}_P$ and $N = N_0 + 1$, we have

$$\text{tr}(P_Y \mathcal{G}^N(|\varphi\rangle\langle\varphi|)) \leq \epsilon,$$

which is

$$\langle \varphi | (\mathcal{G}^*)^N(P_Y) | \varphi \rangle \leq \epsilon.$$

Consider any $|\psi\rangle \in \mathcal{X}$, in the proof of Lemma D.1, we already know that

$$1 = \text{tr}(P_X \mathcal{G}^n(|\psi\rangle\langle\psi|)) \leq \text{tr}((P_X + P_Y) \mathcal{G}^n(|\psi\rangle\langle\psi|)) \leq 1.$$

As the same in the proof of Lemma D.1, we also have

$$|\psi\rangle = (\mathcal{G}^*)^N(P_X)|\psi\rangle = (\mathcal{G}^*)^N(P_X + P_Y)|\psi\rangle.$$

Thus, for any $|\psi\rangle \in \mathcal{X}$, $(\mathcal{G}^*)^N(P_{\mathcal{Y}})|\psi\rangle = 0|\psi\rangle$ (zero vector).

Now, consider any $|\alpha\rangle = a|\psi\rangle + b|\varphi\rangle \in \mathcal{H}_P$, where $|\psi\rangle \in \mathcal{X}$, $|\varphi\rangle \in \mathcal{Y}$, $|a|^2 + |b|^2 = 1$, we have

$$\begin{aligned} \langle \alpha | ((\mathcal{G}^*)^N(P_{\mathcal{Y}}) - \epsilon P_{\mathcal{Y}}) | \alpha \rangle &= a\bar{a}\langle \psi | (\mathcal{G}^*)^N(P_{\mathcal{Y}}) | \psi \rangle + b\bar{b}\langle \varphi | (\mathcal{G}^*)^N(P_{\mathcal{Y}}) | \varphi \rangle - \epsilon b\bar{b} \\ &\quad + \bar{a}b\langle \psi | (\mathcal{G}^*)^N(P_{\mathcal{Y}}) | \varphi \rangle + a\bar{b}\langle \varphi | (\mathcal{G}^*)^N(P_{\mathcal{Y}}) | \psi \rangle \\ &= b\bar{b}\langle \varphi | (\mathcal{G}^*)^N(P_{\mathcal{Y}}) | \varphi \rangle - \epsilon b\bar{b} \\ &\leq \epsilon b\bar{b} - \epsilon b\bar{b} = 0. \end{aligned}$$

Then, $(\mathcal{G}^*)^N(P_{\mathcal{Y}}) \subseteq \epsilon P_{\mathcal{Y}}$. We finally conclude that

$$\forall \epsilon \in (0, 1). \exists N > 0. (\mathcal{G}^*)^N(P_{\mathcal{Y}}) \subseteq \epsilon P_{\mathcal{Y}}.$$

□

PROOF OF LEMMA 5.3. From Lemma D.1, we have $(\mathcal{G}^*)(P_{\mathcal{X}}) \supseteq P_{\mathcal{X}}$, which is

$$\mathcal{E}_1^* \circ \mathcal{E}^*(P_{\mathcal{X}}) \supseteq P_{\mathcal{X}}$$

with $I_{\mathcal{H}_P} \supseteq \mathcal{E}^*(I_{\mathcal{H}_P}) \supseteq \mathcal{E}^*(P_{\mathcal{X}})$, it results $\mathcal{E}_1^*(I_{\mathcal{H}_P}) \supseteq P_{\mathcal{X}}$. Since

$$\mathcal{E}_0^*(I_{\mathcal{H}_P}) + \mathcal{E}_1^*(I_{\mathcal{H}_P}) = I_{\mathcal{H}_P} = P_{\mathcal{X}} + P_{\mathcal{Y}},$$

we have

$$\mathcal{E}_0^*(I_{\mathcal{H}_P}) \subseteq P_{\mathcal{Y}}.$$

By Lemma D.2, for any $\epsilon \in (0, 1)$, there exists $N > 0$ such that $(\mathcal{G}^*)^N(P_{\mathcal{Y}}) \subseteq \epsilon P_{\mathcal{Y}}$. Therefore, for any $n \in \mathbb{N}$, any $\rho \in \mathcal{D}(\mathcal{H}_P)$, we have

$$\begin{aligned} \text{tr}(\mathcal{E}_0 \circ (\mathcal{E} \circ \mathcal{E}_1)^n(\rho)) &= \text{tr}((\mathcal{G}^*)^n(\mathcal{E}_0^*(I_{\mathcal{H}_P}))\rho) \\ &\leq \text{tr}((\mathcal{G}^*)^n(P_{\mathcal{Y}})\rho) \\ &= \text{tr}((\mathcal{G}^*)^{\lfloor \frac{n}{N} \rfloor * N + (n - \lfloor \frac{n}{N} \rfloor * N)}(P_{\mathcal{Y}})\rho) \\ &\leq \epsilon^{\lfloor \frac{n}{N} \rfloor} \text{tr}(P_{\mathcal{Y}}\rho) \\ &\leq \epsilon^{\lfloor \frac{n}{N} \rfloor} \text{tr}(\rho). \end{aligned}$$

□

D.2 Proof of Lemma 5.1

PROOF.

$$\begin{aligned} \frac{d}{d\theta}(\mathcal{E}_2 \circ \mathcal{E}_{H,\theta} \circ \mathcal{E}_1(\rho)) &= \frac{d}{d\theta} \left(\mathcal{E}_2 \left(e^{-i\theta H} \mathcal{E}_1(\rho) e^{i\theta H} \right) \right) \\ &= \mathcal{E}_2 \left(-iHe^{-i\theta H} \mathcal{E}_1(\rho) e^{i\theta H} + e^{-i\theta H} \mathcal{E}_1(\rho) e^{i\theta H} (iH) \right) \\ &= \mathcal{E}_2 \left(e^{-i\theta H} (-iH \mathcal{E}_1(\rho)) e^{i\theta H} + e^{-i\theta H} (\mathcal{E}_1(\rho) (iH)) e^{i\theta H} \right) \\ &= \mathcal{E}_2 \circ \mathcal{E}_{H,\theta} (-iH \mathcal{E}_1(\rho) + i\mathcal{E}_1(\rho) H) \\ &= \mathcal{E}_2 \circ \mathcal{E}_{H,\theta} (-i[H, \mathcal{E}_1(\rho)]). \end{aligned}$$

□

D.3 Proof of Commutator-form Rule

The following lemma modified from **qPCA (quantum principal component analysis)** [44] helps incorporate commutators into the semantics of parameterized quantum programs:

LEMMA D.3 (MODIFIED FROM REFERENCE [44]). *Let $\mathcal{H}_1, \mathcal{H}_2, \mathcal{H}_3$ be Hilbert spaces with $\dim(\mathcal{H}_2) = \dim(\mathcal{H}_3)$, S a SWAP operator on $\mathcal{H}_2 \otimes \mathcal{H}_3$, $\rho \in \mathcal{D}(\mathcal{H}_1 \otimes \mathcal{H}_2)$, $\sigma \in \mathcal{D}(\mathcal{H}_3)$, and parameter $\alpha \in \mathbb{R}$. Then,*

$$\text{tr}_3(e^{-i\alpha S} \rho \otimes \sigma e^{i\alpha S}) = \cos(\alpha)^2 \rho + \sin(\alpha)^2 \text{tr}_2(\rho) \otimes \sigma - i \cos(\alpha) \sin(\alpha) [[\sigma]_2, \rho], \quad (\text{D.1})$$

where $[\sigma]_2$ denotes the operator $I_1 \otimes \sigma$ on $\mathcal{H}_1 \otimes \mathcal{H}_2$ with I_1 being the identity operator on $\mathcal{H}_1$, and tr_j denotes the partial trace on $\mathcal{H}_j$.

PROOF.

$$\begin{aligned} & \text{tr}_3(e^{-i\alpha S} \rho \otimes \sigma e^{i\alpha S}) \\ &= \text{tr}_3((\cos(\alpha)I - i \sin(\alpha)S)\rho \otimes \sigma(\cos(\alpha)I + i \sin(\alpha)S)) \\ &= \text{tr}_3\left(\cos(\alpha)^2 \rho \otimes \sigma + \sin(\alpha)^2 S\rho \otimes \sigma S + i \cos(\alpha) \sin(\alpha) \rho \otimes \sigma S - i \cos(\alpha) \sin(\alpha) S\rho \otimes \sigma\right) \\ &= \cos(\alpha)^2 \rho + \sin(\alpha)^2 \text{tr}_2(\rho) \otimes \sigma + i \cos(\alpha) \sin(\alpha) \rho[\sigma]_2 - i \cos(\alpha) \sin(\alpha) [\sigma]_2 \rho \\ &= \cos(\alpha)^2 \rho + \sin(\alpha)^2 \text{tr}_2(\rho) \otimes \sigma - i \cos(\alpha) \sin(\alpha) [[\sigma]_2, \rho]. \end{aligned}$$

□

With Lemma D.3, the $g(\theta; \alpha)$ in Equation (5.2) can be rewritten as

$$\begin{aligned} g(\theta; \alpha) &= \cos(\alpha)^2 \text{tr}\left(\mathcal{O}\mathcal{E}_2\left(e^{-i\theta\sigma}\mathcal{E}_1(\rho)e^{i\theta\sigma}\right)\right) + \cos(\alpha)^2 \text{tr}\left(\mathcal{O}\mathcal{E}_2\left(e^{-i\theta\sigma}(\sigma \otimes \text{tr}_1(\mathcal{E}_1(\rho)))e^{i\theta\sigma}\right)\right) \\ &\quad + \cos(\alpha) \sin(\alpha) \text{tr}\left(\mathcal{O}\mathcal{E}_2\left(e^{-i\theta\sigma}(-i[\sigma \otimes I, \mathcal{E}_1(\rho)])e^{i\theta\sigma}\right)\right), \end{aligned}$$

where $\text{tr}_1(\mathcal{E}_1(\rho))$ is partial trace of $\mathcal{E}_1(\rho)$ over the space of $e^{-i\theta\sigma}$ acts. Thus,

$$g(\theta; \alpha) - g(\theta; -\alpha) = 2 \cos(\alpha) \sin(\alpha) \text{tr}\left(\mathcal{O}\mathcal{E}_2\left(e^{-i\theta\sigma}(-i[\sigma \otimes I, \mathcal{E}_1(\rho)])e^{i\theta\sigma}\right)\right) = \sin(2\alpha) \frac{d}{d\theta} f(\theta).$$

D.4 Proof of Theorem 5.4

To prove Theorem 5.4, we need the Super-operator-valued Transition Systems [74], which provide us with a convenient way for modeling the control flow of quantum programs. In there, we use a modified version.

Definition D.4 (Modified Super-operator-valued Transition Systems). A **modified super-operator-valued transition system (mSVTS)** is a 5-tuple $S = \langle \mathcal{H}, L, l_0, \mathcal{T}, \rho_0 \rangle$, where:

- $\mathcal{H}$ is a Hilbert space called the state space;
- L is a finite set of locations;
- $l_0 \in L$ is the initial location;
- $\mathcal{T}$ is a set of transitions. Each transition $\tau \in \mathcal{T}$ is a triple $\tau = \langle l, l', \mathcal{E} \rangle$, often written as $\tau = l \xrightarrow{\mathcal{E}} l'$, where $l, l' \in L$ are pre- and post-locations of τ , respectively, and $\mathcal{E}$ is a super-operator in $\mathcal{H}$. It is required that

$$\sum \left\{ \mathcal{E}^*(I_{\mathcal{H}}) : l \xrightarrow{\mathcal{E}} l' \in \mathcal{T} \right\} \subseteq I_{\mathcal{H}} \quad (\text{D.2})$$

for each $l \in L$, where $I_{\mathcal{H}}$ is the identify operator on $\mathcal{H}$ and $\mathcal{E}^*$ is the Schrödinger-Heisenberg dual of $\mathcal{E}$.

— ρ_0 is an initial state at l_0 .

For any path $\pi = l_1 \xrightarrow{\mathcal{E}_1} l_2 \xrightarrow{\mathcal{E}_2} \dots \xrightarrow{\mathcal{E}_{n-1}} l_n$ in the mSVTS graph, we write $l_1 \xRightarrow{\pi} l_n$ and use $\mathcal{E}_\pi$ to denote the composition of the super-operator along the path, i.e., $\mathcal{E}_\pi = \mathcal{E}_{n-1} \circ \dots \circ \mathcal{E}_2 \circ \mathcal{E}_1$. If the transition $l \xrightarrow{\mathcal{E}} l'$ in $\mathcal{T}$ has superoperator $\mathcal{E}$ simply defined by an operator E , i.e., $\mathcal{E}(\rho) = E\rho E^\dagger$ for all density operator ρ in $\mathcal{H}$, then we will write $l \xrightarrow{E} l'$ for $l \xrightarrow{\mathcal{E}} l'$.

LEMMA D.5. *Let A be a set of paths in $\mathcal{S} = \langle \mathcal{H}, L, l_0, \mathcal{T}, \rho_0 \rangle$. All paths in A have a same initial location and each path $\pi \in A$ is not a prefix of others in A , then for any $\rho \in \mathcal{D}(\mathcal{H})$,*

$$\sum_{\pi \in A} \text{tr}(\mathcal{E}_\pi(\rho)) \leq \text{tr}(\rho).$$

PROOF. We first assume that A is finite and prove it by induction through the size of $|A|$.

— $|A| = 1$. A has only one element π . We write $\pi = \pi = l_1 \xrightarrow{\mathcal{E}_1} l_2 \xrightarrow{\mathcal{E}_2} \dots \xrightarrow{\mathcal{E}_{n-1}} l_n$, then by Formula (D.2), for any $\rho \in \mathcal{D}(\mathcal{H})$,

$$\begin{aligned} \text{tr}(\mathcal{E}_\pi(\rho)) &= \text{tr}(\mathcal{E}_{n-1} \circ \dots \circ \mathcal{E}_2 \circ \mathcal{E}_1(\rho)) \\ &= \text{tr}(\mathcal{E}_{n-1}^*(I_{\mathcal{H}}) \cdot \mathcal{E}_{n-2} \circ \dots \circ \mathcal{E}_2 \circ \mathcal{E}_1(\rho)) \\ &\leq \text{tr}(\mathcal{E}_{n-2} \circ \dots \circ \mathcal{E}_2 \circ \mathcal{E}_1(\rho)) \\ &\dots \\ &\leq \text{tr}(\rho). \end{aligned}$$

If $|A| = 0$, then $\sum \pi \in A \text{tr}(\mathcal{E}_\pi(\rho)) = 0 \leq \text{tr}(\rho)$.

— Suppose when $|A| \leq n, n \geq 1$, we have that for any $\rho \in \mathcal{D}(\mathcal{H})$,

$$\sum_{\pi \in A} \text{tr}(\mathcal{E}_\pi(\rho)) \leq \text{tr}(\rho).$$

Then, consider $|A| = n + 1$, we choose a path $\pi = l_1 \xrightarrow{\mathcal{E}_1} l_2 \xrightarrow{\mathcal{E}_2} \dots \xrightarrow{\mathcal{E}_{n-1}} l_n \in A$ and let $\pi_j = l_j \xrightarrow{\mathcal{E}_j} l_{j+1}, 1 \leq j \leq n - 1$, then $\pi = \pi_1 \pi_2 \dots \pi_{n-1}$. For convenience, we use π_0 to denote an empty path. Then, for this π , we define

$$B = \{j : 0 \leq j \leq n - 1, \forall \pi \in A. \exists \pi'. \text{s.t. } \pi = \pi_0 \pi_1 \pi_2 \dots \pi_j \pi'\}.$$

B must contain 0, thus B is not empty. As each path in A is not a prefix of others in A , we have that $n - 1 \notin B$. Let $j_0 = \max B$, then $j_0 < n - 1$. Consider all the transitions in $\mathcal{T}$ with l_{j_0} as pre-location:

$$\tau_1 = l_{j_0} \xrightarrow{\mathcal{G}_1} l'_1, \tau_2 = l_{j_0} \xrightarrow{\mathcal{G}_2} l'_2, \dots, \tau_{n'} = l_{j_0} \xrightarrow{\mathcal{G}_{n'}} l'_{n'}.$$

It is followed that $\forall \pi' \in A$, π' must have a prefix $\pi_0 \pi_1 \dots \pi_{j_0} \tau_k$ with $1 \leq k \leq n'$, otherwise $\pi' = \pi_0 \pi_1 \dots \pi_{j_0}$ (if $j_0 > 0$, then $\pi' \neq \pi_0$; if $j_0 = 0$, then this π' does not exist), it is a prefix for all paths in A , which is a contradiction. Therefore,

$$A = \bigcup_{k=1}^{n'} C_k$$

with

$$C_k \equiv \{\pi' \in A : \exists \pi''. \text{s.t. } \pi' = \pi_0 \pi_1 \dots \pi_{j_0} \tau_k \pi''\},$$

for $1 \leq k \leq n'$, and each path in C_k is not a prefix of others in C_k . We claim that $|C_k| \leq n$, otherwise there exists $C_{k_0} = A$, then $\pi_0 \pi_1 \cdots \pi_{j_0} \tau_{k_0}$ is a prefix for all paths in A , this contradicts the definition of j_0 , because in that case $j_0 + 1 \in B$. Then, define

$$D_k \equiv \{\pi'' : \pi_0 \pi_1 \cdots \pi_{j_0} \tau_k \pi'' \in C_k\}, \quad 1 \leq k \leq n',$$

we have that for any $1 \leq k \leq n'$, all paths in D_k have same initial location and each path $\pi'' \in D_k$ is not a prefix of others in D_k and $|D_k| = |C_k| \leq n$. By inductive hypothesis, for any $\rho \in \mathcal{D}(\mathcal{H})$,

$$\sum_{\pi'' \in D_k} \text{tr}(\mathcal{E}_{\pi''}(\rho)) \leq \text{tr}(\rho), \quad 1 \leq k \leq n'. \quad (\text{D.3})$$

Finally, for any $\rho \in \mathcal{D}(\mathcal{H})$,

$$\begin{aligned} \sum_{\pi' \in A} \text{tr}(\mathcal{E}_{\pi'}(\rho)) &= \sum_{k=1}^{n'} \sum_{\pi' \in C_k} \text{tr}(\mathcal{E}_{\pi'}(\rho)) \\ &= \sum_{k=1}^{n'} \sum_{\pi'' \in D_k} \text{tr}(\mathcal{E}_{\pi_0 \pi_1 \cdots \pi_{j_0} \tau_k \pi''}(\rho)) \\ &= \sum_{k=1}^{n'} \sum_{\pi'' \in D_k} \text{tr}(\mathcal{E}_{\pi''}(\mathcal{E}_{\pi_0 \pi_1 \cdots \pi_{j_0} \tau_k}(\rho))) \\ &\leq \sum_{k=1}^{n'} \text{tr}(\mathcal{E}_{\pi_0 \pi_1 \cdots \pi_{j_0} \tau_k}(\rho)) \quad (\text{by Inequality (D.3)}) \\ &= \sum_{k=1}^{n'} \text{tr}(\mathcal{E}_{\tau_k}(\mathcal{E}_{\pi_0 \pi_1 \cdots \pi_{j_0}}(\rho))) \\ &= \sum_{k=1}^{n'} \text{tr}(\mathcal{G}_k(\mathcal{E}_{\pi_0 \pi_1 \cdots \pi_{j_0}}(\rho))) \\ &\leq \text{tr}(\mathcal{E}_{\pi_0 \pi_1 \cdots \pi_{j_0}}(\rho)) \quad (\text{by Formula (D.2)}) \\ &\leq \text{tr}(\rho). \quad (\text{as the same proof of } |A| = 1) \end{aligned}$$

Thus, when $|A|$ is finite, we prove this proposition. Because of the order-preserving property of limitation, when $|A| = \infty$, we also have the same result. $\square$

Definition D.6 (Computation Path of mSVTS). A path $\pi = l_0 \xrightarrow{\mathcal{E}_1} m_1 \xrightarrow{\mathcal{E}_2} \cdots \xrightarrow{\mathcal{E}_n} m_n$ in $\mathcal{S} = \langle \mathcal{H}, L, l_0, \mathcal{T}, \rho_0 \rangle$ is a computation path if for any τ in $\mathcal{T}$, m_n is not a pre-location of τ . And we write $\Pi_{\mathcal{S}}$ be the set of computation paths, which is

$$\Pi_{\mathcal{S}} = \{\pi \text{ is a path in } \mathcal{S} \mid l_0 \xRightarrow{\pi} l \wedge \forall \tau \in \mathcal{T}, l \text{ is not a pre-location of } \tau\}.$$

Moreover, we use $\Pi_{\mathcal{S}}^{(n)}$ to denote the set of length $\leq n$ (transits $\leq n$ steps) paths in $\Pi_{\mathcal{S}}$.

As similar in Reference [74], the control flow graph of a quantum program can be represented by an mSVTS. For every parameterized quantum **while**-program $P(\theta)$, we define an mSVTS $\mathcal{S}_{P(\theta)} = \langle \mathcal{H}_{P(\theta)}, L, l_{in}^{P(\theta)}, \mathcal{T}, \rho \rangle$ in the state Hilbert space $\mathcal{H}_{P(\theta)}$ of $P(\theta)$ by induction on the program structure of $P(\theta)$, where ρ is an input state of $P(\theta)$. This transition system has two designated locations l_{in}^P, l_{out}^P , with the former being the initial location and the latter being the exit location. We only need to consider definitions of L and $\mathcal{T}$.

- $P(\theta) \equiv \text{skip}$. $\mathcal{S}_{P(\theta)}$ has only two locations $l_{in}^{P(\theta)}, l_{out}^{P(\theta)}$ and a single transition $l_{in}^{P(\theta)} \xrightarrow{I} l_{out}^{P(\theta)}$.
- $P(\theta) \equiv q := |0\rangle$. Let $\{|n\rangle_q\}$ be the basis of $\mathcal{H}_q$, then $\mathcal{S}_{P_\theta}$ has two locations $l_{in}^{P(\theta)}, l_{out}^{P(\theta)}$ and a single transition $l_{in}^{P(\theta)} \xrightarrow{\mathcal{E}_q} l_{out}^{P(\theta)}$, where $\mathcal{E}_q(\rho) = \sum_n |0\rangle_q \langle n| \rho |n\rangle_q \langle 0|$.
- $P(\theta) \equiv \bar{q} := \sigma$.⁹ Write σ in spectral decomposition, $\sigma = \sum_n \lambda_n |\psi_n\rangle_{\bar{q}} \langle \psi_n|$, $\lambda_n \geq 0$ (there, we include eigenvalues of 0), then $\mathcal{S}_{P_\theta}$ has two locations $l_{in}^{P(\theta)}, l_{out}^{P(\theta)}$ and a single transition $l_{in}^{P(\theta)} \xrightarrow{\mathcal{E}_{\bar{q}, \sigma}} l_{out}^{P(\theta)}$, where

$$\mathcal{E}_{\bar{q}, \sigma}(\rho) = \sum_{m, n} (\sqrt{\lambda_m} |\psi_m\rangle_{\bar{q}} \langle \psi_n|) \rho (\sqrt{\lambda_m} |\psi_n\rangle_{\bar{q}} \langle \psi_m|).$$

- $P(\theta) \equiv \bar{q} := U[\bar{q}]$. $\mathcal{S}_{P(\theta)}$ has two locations $l_{in}^{P(\theta)}, l_{out}^{P(\theta)}$ and a single transition $l_{in}^{P(\theta)} \xrightarrow{[U]_{\bar{q}}} l_{out}^{P(\theta)}$.
- $P(\theta) \equiv \bar{q} := e^{-i\theta\sigma}[\bar{q}]$. $\mathcal{S}_{P(\theta)}$ has two locations $l_{in}^{P(\theta)}, l_{out}^{P(\theta)}$ and a single transition $l_{in}^{P(\theta)} \xrightarrow{[e^{-i\theta\sigma}]_{\bar{q}}} l_{out}^{P(\theta)}$.
- $P(\theta) \equiv P_1(\theta); P_2(\theta)$. Suppose $\mathcal{S}_{P_1(\theta)}, \mathcal{S}_{P_2(\theta)}$ are the control flow graphs of subprograms $P_1(\theta), P_2(\theta)$, respectively. Then, $\mathcal{S}_{P(\theta)}$ is constructed as follows: We identify $l_{out}^{P_1(\theta)} = l_{in}^{P_2(\theta)}$ and concatenate $\mathcal{S}_{P_1(\theta)}, \mathcal{S}_{P_2(\theta)}$. We further set $l_{in}^{P_1(\theta)} = l_{in}^{P(\theta)}, l_{out}^{P_2(\theta)} = l_{out}^{P(\theta)}$.
- $P(\theta) \equiv \text{if } (\Box m \cdot M[\bar{q}] = m \rightarrow P_m(\theta)) \text{ fi}$. Suppose that $\mathcal{S}_{P_m(\theta)}$ is the control flow graph of subprogram $P_m(\theta)$ for every m . Then, $\mathcal{S}_{P(\theta)}$ is constructed as follows: We put all $\mathcal{S}_{P_m(\theta)}$'s together and add a new location $l_{in}^{P(\theta)}$ and a transition $l_{in}^{P(\theta)} \xrightarrow{[M]_{\bar{q}}} l_{in}^{P_m(\theta)}$ for every m . We further set $l_{out}^{P_m(\theta)} = l_{out}^{P(\theta)}$ for all m .
- $P(\theta) \equiv \text{while } M[\bar{q}] = 1 \text{ do } Q(\theta) \text{ od}$. We construct $\mathcal{S}_{P(\theta)}$ from the control flow graph $\mathcal{S}_{Q(\theta)}$ of subprogram $Q(\theta)$ as follows: We add two new locations $l_{in}^{P(\theta)}, l_{out}^{P(\theta)}$ and two transitions $l_{in}^{P(\theta)} \xrightarrow{[M_0]_{\bar{q}}} l_{out}^{P(\theta)}, l_{in}^{P(\theta)} \xrightarrow{[M_1]_{\bar{q}}} l_{in}^{Q(\theta)}$. We further identify $l_{out}^{Q(\theta)} = l_{in}^{P(\theta)}$.

There, we use the subscript $[U]_{\bar{q}}$ for unitary U and quantum variables $\bar{q}$ to indicate that U acts on the Hilbert space $\mathcal{H}_{\bar{q}}$.

THEOREM D.7. *For a parameterized quantum **while**-program $P(\theta)$ with an initial state ρ and its mSVTS $\mathcal{S}_{P(\theta)}$, we have*

$$\llbracket P(\theta) \rrbracket(\rho) = \sum_{\pi \in \Pi_{\mathcal{S}_{P(\theta)}}} \mathcal{E}_\pi(\rho) \equiv \bigsqcup_{n=1}^{\infty} \sum_{\pi \in \Pi_{\mathcal{S}_{P(\theta)}^{(n)}}} \mathcal{E}_\pi(\rho).$$

PROOF. We prove it by induction through the program structure.

- $P(\theta) \equiv \text{skip}$. We have $\Pi_{\mathcal{S}_{P(\theta)}} = \{l_{in}^{P(\theta)} \xrightarrow{I} l_{out}^{P(\theta)}\}$, then

$$\sum_{\pi \in \Pi_{\mathcal{S}_{P(\theta)}}} \mathcal{E}_\pi(\rho) = I\rho I = \rho = \llbracket \text{skip} \rrbracket(\rho).$$

- $P(\theta) \equiv q := |0\rangle$. We have $\Pi_{\mathcal{S}_{P(\theta)}} = \{l_{in}^{P(\theta)} \xrightarrow{\mathcal{E}_q} l_{out}^{P(\theta)}\}$, then

$$\sum_{\pi \in \Pi_{\mathcal{S}_{P(\theta)}}} \mathcal{E}_\pi(\rho) = \mathcal{E}_q(\rho) = \sum_n |0\rangle_q \langle n| \rho |n\rangle_q \langle 0| = \llbracket q := |0\rangle \rrbracket(\rho).$$

⁹Although this statement is not contained in the syntax, we use it for convenience, as we have said in Remark 4.2.

– $P(\theta) \equiv \bar{q} := \sigma$. We have $\Pi_{S_{P(\theta)}} = \{l_{in}^{P(\theta)} \xrightarrow{\mathcal{E}_{\bar{q}, \sigma}} l_{out}^{P(\theta)}\}$, then

$$\sum_{\pi \in \Pi_{S_{P(\theta)}}} \mathcal{E}_{\pi}(\rho) = \mathcal{E}_{\bar{q}, \sigma}(\rho) = \sum_{m, n} (\sqrt{\lambda_m} |\psi_m\rangle_{\bar{q}} \langle \psi_n|) \rho (\sqrt{\lambda_m} |\psi_n\rangle_{\bar{q}} \langle \psi_m|).$$

In the Remark 4.2, the statement $\bar{q} := \sigma$ aims to set the state ρ to $\text{tr}_{\bar{q}}(\rho) \otimes \sigma$, which should be $\llbracket \bar{q} := \sigma \rrbracket(\rho)$. We can check that for any ρ

$$\sum_{m, n} (\sqrt{\lambda_m} |\psi_m\rangle_{\bar{q}} \langle \psi_n|) \rho (\sqrt{\lambda_m} |\psi_n\rangle_{\bar{q}} \langle \psi_m|) = \text{tr}_{\bar{q}}(\rho) \otimes \sigma.$$

Thus,

$$\sum_{\pi \in \Pi_{S_{P(\theta)}}} \mathcal{E}_{\pi}(\rho) = \llbracket \bar{q} := \sigma \rrbracket(\rho).$$

– $P(\theta) \equiv \bar{q} := U[\bar{q}]$. We have $\Pi_{S_{P(\theta)}} = \{l_{in}^{P(\theta)} \xrightarrow{[U]_{\bar{q}}} l_{out}^{P(\theta)}\}$, then

$$\sum_{\pi \in \Pi_{S_{P(\theta)}}} \mathcal{E}_{\pi}(\rho) = [U]_{\bar{q}} \rho [U^{\dagger}]_{\bar{q}} = \llbracket \bar{q} := U[\bar{q}] \rrbracket(\rho).$$

– $P(\theta) \equiv \bar{q} := e^{-i\theta\sigma}[\bar{q}]$. We have $\Pi_{S_{P(\theta)}} = \{l_{in}^{P(\theta)} \xrightarrow{[e^{-i\theta\sigma}]_{\bar{q}}} l_{out}^{P(\theta)}\}$, then

$$\sum_{\pi \in \Pi_{S_{P(\theta)}}} \mathcal{E}_{\pi}(\rho) = [e^{-i\theta\sigma}]_{\bar{q}} \rho [e^{i\theta\sigma}]_{\bar{q}} = \llbracket \bar{q} := e^{-i\theta\sigma}[\bar{q}] \rrbracket(\rho).$$

– $P(\theta) \equiv P_1(\theta); P_2(\theta)$. For any $\pi_1 \in \Pi_{S_{P_1(\theta)}}$, $\pi_2 \in \Pi_{S_{P_2(\theta)}}$, we can write

$$\begin{aligned} \pi_1 &= l_{in}^{P_1(\theta)} \xrightarrow{\mathcal{E}_1^{P_1(\theta)}} l_1^{P_1(\theta)} \xrightarrow{\mathcal{E}_2^{P_1(\theta)}} \dots \xrightarrow{\mathcal{E}_m^{P_1(\theta)}} l_{out}^{P_1(\theta)} \\ \pi_2 &= l_{in}^{P_2(\theta)} \xrightarrow{\mathcal{E}_1^{P_2(\theta)}} l_1^{P_2(\theta)} \xrightarrow{\mathcal{E}_2^{P_2(\theta)}} \dots \xrightarrow{\mathcal{E}_n^{P_2(\theta)}} l_{out}^{P_2(\theta)}, \end{aligned}$$

then, from our construction

$$\begin{aligned} \pi &= l_{in}^{P(\theta)} \xrightarrow{\mathcal{E}_1^{P_1(\theta)}} l_1^{P_1(\theta)} \xrightarrow{\mathcal{E}_2^{P_1(\theta)}} \dots \xrightarrow{\mathcal{E}_m^{P_1(\theta)}} l_{out}^{P_1(\theta)} \\ &\quad \xrightarrow{\mathcal{E}_1^{P_2(\theta)}} l_1^{P_2(\theta)} \xrightarrow{\mathcal{E}_2^{P_2(\theta)}} \dots \xrightarrow{\mathcal{E}_n^{P_2(\theta)}} l_{out}^{P_2(\theta)} \in \Pi_{S_{P(\theta)}}. \end{aligned}$$

For convenience, we write $\pi = \pi_1 \pi_2$, then

$$\{\pi_1 \pi_2 \mid \pi_1 \in \Pi_{S_{P_1(\theta)}}, \pi_2 \in \Pi_{S_{P_2(\theta)}}\} \subseteq \Pi_{S_{P(\theta)}}.$$

However, for any $\pi \in \Pi_{S_{P(\theta)}}$, write

$$\pi = l_{in}^{P(\theta)} \xrightarrow{\mathcal{E}_1^{P(\theta)}} l_1^{P(\theta)} \xrightarrow{\mathcal{E}_2^{P(\theta)}} \dots \xrightarrow{\mathcal{E}_m^{P(\theta)}} l_{out}^{P(\theta)}.$$

From the construction, we have that $l_{in}^{P(\theta)} = l_{in}^{P_1(\theta)}$, $l_{out}^{P(\theta)} = l_{out}^{P_2(\theta)}$. Then, we can define k to be the first index such that $l_k^{P(\theta)}$ is in $S_{P_1(\theta)}$ and $l_{k+1}^{P(\theta)}$ is in $S_{P_2(\theta)}$. Moreover, for any location $l \neq l_{out}^{P_1(\theta)}$ in $S_{P_1(\theta)}$, its post-location is still in $S_{P_1(\theta)}$, then we have $l_k^{P(\theta)} = l_{out}^{P_1(\theta)}$. By construction, $l_{in}^{P_2(\theta)} = l_{out}^{P_1(\theta)} = l_k^{P(\theta)}$ and for any location l in $S_{P_2(\theta)}$, its post-location is still in $S_{P_2(\theta)}$, thus for all $j \geq k+1$, $l_j^{P(\theta)}$ is in $S_{P_2(\theta)}$. Then,

$$\pi_1 = l_{in}^{P(\theta)} \xrightarrow{\mathcal{E}_1^{P(\theta)}} l_1^{P(\theta)} \xrightarrow{\mathcal{E}_2^{P(\theta)}} \dots \xrightarrow{\mathcal{E}_k^{P(\theta)}} l_k^{P(\theta)} \in \Pi_{S_{P_1(\theta)}}$$

$$\pi_2 = l_k^{P(\theta)} \xrightarrow{\mathcal{E}_{k+1}^{P(\theta)}} l_{k+1}^{P(\theta)} \xrightarrow{\mathcal{E}_{k+1}^{P(\theta)}} \dots \xrightarrow{\mathcal{E}_k^{P(\theta)}} l_m^{P(\theta)} \in \Pi_{S_{P_2}(\theta)},$$

therefore,

$$\{\pi_1 \pi_2 \mid \pi_1 \in \Pi_{S_{P_1}(\theta)}, \pi_2 \in \Pi_{S_{P_2}(\theta)}\} \supseteq \Pi_{S_P(\theta)}.$$

Because all $\mathcal{E}_\pi(\rho)$ are positive semidefinite, we consider the partial summation, for any $n, m \geq 1$:

$$\sum_{\pi \in \Pi_{S_P(\theta)}^{(n)}} \mathcal{E}_\pi(\rho) \sqsubseteq \sum_{\pi \in \Pi_{S_{P_2}(\theta)}^{(n)}} \sum_{\pi \in \Pi_{S_{P_1}(\theta)}^{(n)}} \mathcal{E}_{\pi_1 \pi_2}(\rho) = \sum_{\pi \in \Pi_{S_{P_2}(\theta)}^{(n)}} \mathcal{E}_{\pi_2} \left(\sum_{\pi \in \Pi_{S_{P_1}(\theta)}^{(n)}} \mathcal{E}_{\pi_1}(\rho) \right) \quad (\text{D.4})$$

$$\sum_{\pi \in \Pi_{S_P(\theta)}^{(m+n)}} \mathcal{E}_\pi(\rho) \sqsupseteq \sum_{\pi \in \Pi_{S_{P_2}(\theta)}^{(m)}} \sum_{\pi \in \Pi_{S_{P_1}(\theta)}^{(n)}} \mathcal{E}_{\pi_1 \pi_2}(\rho) = \sum_{\pi \in \Pi_{S_{P_2}(\theta)}^{(m)}} \mathcal{E}_{\pi_2} \left(\sum_{\pi \in \Pi_{S_{P_1}(\theta)}^{(n)}} \mathcal{E}_{\pi_1}(\rho) \right). \quad (\text{D.5})$$

By the inductive hypothesis, for any σ :

$$\begin{aligned} \sum_{\pi \in \Pi_{S_{P_1}(\theta)}} \mathcal{E}_\pi(\sigma) &= \llbracket P_1(\theta) \rrbracket(\sigma) \\ \sum_{\pi \in \Pi_{S_{P_2}(\theta)}} \mathcal{E}_\pi(\sigma) &= \llbracket P_2(\theta) \rrbracket(\sigma). \end{aligned}$$

Therefore, in Equation (D.4), we have

$$\begin{aligned} \sum_{\pi \in \Pi_{S_P(\theta)}^{(n)}} \mathcal{E}_\pi(\rho) &\sqsubseteq \sum_{\pi \in \Pi_{S_{P_2}(\theta)}^{(n)}} \mathcal{E}_{\pi_2} \left(\sum_{\pi \in \Pi_{S_{P_1}(\theta)}^{(n)}} \mathcal{E}_{\pi_1}(\rho) \right) \\ &\sqsubseteq \sum_{\pi \in \Pi_{S_{P_2}(\theta)}^{(n)}} \mathcal{E}_{\pi_2}(\llbracket P_1(\theta) \rrbracket(\rho)) \\ &\sqsubseteq \llbracket P_2(\theta) \rrbracket(\llbracket P_1(\theta) \rrbracket(\rho)) \\ &= \llbracket P_1(\theta); P_2(\theta) \rrbracket(\rho) \end{aligned}$$

and, in Equation (D.5), let $n \rightarrow \infty$, we have for any $m \geq 1$:

$$\sum_{\pi \in \Pi_{S_P(\theta)}} \mathcal{E}_\pi(\rho) \sqsupseteq \sum_{\pi \in \Pi_{S_{P_2}(\theta)}^{(m)}} \mathcal{E}_{\pi_2}(\llbracket P_1(\theta) \rrbracket(\rho))$$

then, let $m \rightarrow \infty$, we get

$$\sum_{\pi \in \Pi_{S_P(\theta)}} \mathcal{E}_\pi(\rho) \sqsupseteq \llbracket P_2(\theta) \rrbracket(\llbracket P_1(\theta) \rrbracket(\rho)) = \llbracket P_1(\theta); P_2(\theta) \rrbracket(\rho).$$

Thus, $\sum_{\pi \in \Pi_{S_P(\theta)}} \mathcal{E}_\pi(\rho) = \llbracket P_1(\theta); P_2(\theta) \rrbracket(\rho)$.

– $P(\theta) \equiv \text{if } (\Box m \cdot M[\bar{q}] = m \rightarrow P_m(\theta)) \text{ fi}$. Let $\pi_m = l_{in}^{P(\theta)} \xrightarrow{[M_m]_{\bar{q}}} l_{in}^{P_m(\theta)}$, then

$$\Pi_{S_P(\theta)} = \bigcup_m \{\pi_m \pi \mid \pi \in \Pi_{S_{P_m}(\theta)}\},$$

therefore,

$$\begin{aligned}
\sum_{\pi \in \Pi_{S_P(\theta)}} \mathcal{E}_\pi(\rho) &= \sum_m \sum_{\pi \in \Pi_{S_{P_m}(\theta)}} \mathcal{E}_{\pi_m \pi}(\rho) \\
&= \sum_m \sum_{\pi \in \Pi_{S_{P_m}(\theta)}} \mathcal{E}_\pi(\mathcal{E}_{\pi_m}(\rho)) \\
&= \sum_m \sum_{\pi \in \Pi_{S_{P_m}(\theta)}} \mathcal{E}_\pi([M_m]_{\bar{q}} \rho [M_m^\dagger]_{\bar{q}}) \\
&= \sum_m \llbracket P_m(\theta) \rrbracket ([M_m]_{\bar{q}} \rho [M_m^\dagger]_{\bar{q}}) \\
&= \llbracket \text{if } (\Box m \cdot M[\bar{q}] = m \rightarrow P_m(\theta)) \text{ fi} \rrbracket (\rho).
\end{aligned}$$

In there, the second-to-last equality is provided by the inductive hypothesis.

– $P(\theta) \equiv \text{while } M[\bar{q}] = 1 \text{ do } Q(\theta) \text{ od}$. Let $\pi_0 = l_{in}^{P(\theta)} \xrightarrow{[M_0]_{\bar{q}}} l_{out}^{P(\theta)}$, $\pi_1 = l_{in}^{P(\theta)} \xrightarrow{[M_1]_{\bar{q}}} l_{in}^{Q(\theta)}$. As same before, we can obtain that any path in $\Pi_{S_P(\theta)}$ has the form η_0 or $\pi_1 \eta_1 \pi_1 \eta_2 \cdots \pi_1 \eta_n \pi_0$, $\eta_j \in S_{Q(\theta)}$, $j = 1, 2, \dots, n$, which is

$$\Pi_{S_P(\theta)} = \{\pi_0\} \cup \left\{ \pi_1 \eta_1 \pi_1 \eta_2 \cdots \pi_1 \eta_n \pi_0 \mid (n \in \mathbb{N}_+) \wedge (\forall 1 \leq j \leq n. \eta_j \in \Pi_{S_{Q(\theta)}}) \right\},$$

then, for any $m, n, n_j \geq 1, j = 1, 2, \dots, m$:

$$\Pi_{S_P(\theta)}^{(mn)} \subseteq \{\pi_0\} \cup \left\{ \pi_1 \eta_1 \pi_1 \eta_2 \cdots \pi_1 \eta_k \pi_0 \mid (k \in \mathbb{N}^+, k \leq mn) \wedge (\forall 1 \leq l \leq k. \eta_l \in \Pi_{S_{Q(\theta)}}^{(mn)}) \right\}$$

$$\Pi_{S_P(\theta)}^{((\sum_{j=1}^m n_j) + m + 1)} \supseteq \{\pi_0\} \cup \left\{ \pi_1 \eta_1 \pi_1 \eta_2 \cdots \pi_1 \eta_k \pi_0 \mid (k \in \mathbb{N}^+, k \leq m) \wedge (\forall 1 \leq l \leq k. \eta_l \in \Pi_{S_{Q(\theta)}}^{(n_l)}) \right\}$$

Thus,

$$\begin{aligned}
&\sum_{\pi \in \Pi_{S_P(\theta)}^{(mn)}} \mathcal{E}_\pi(\rho) \\
&\sqsubseteq \mathcal{E}_{\pi_0}(\rho) + \sum_{k=1}^{mn} \sum_{\eta_1, \eta_2, \dots, \eta_k \in \Pi_{S_{Q(\theta)}}^{(mn)}} \mathcal{E}_{\pi_1 \eta_1 \pi_1 \eta_2 \cdots \pi_1 \eta_k \pi_0}(\rho) \\
&= \mathcal{E}_{\pi_0}(\rho) + \sum_{k=1}^{mn} \mathcal{E}_{\pi_0} \left(\sum_{\eta_k \in \Pi_{S_{Q(\theta)}}^{(mn)}} \mathcal{E}_{\eta_k} \circ \mathcal{E}_{\pi_1} \left(\cdots \left(\sum_{\eta_1 \in \Pi_{S_{Q(\theta)}}^{(mn)}} \mathcal{E}_{\eta_1} \circ \mathcal{E}_{\pi_1}(\rho) \right) \cdots \right) \right) \\
&\sqsubseteq \mathcal{E}_{\pi_0}(\rho) + \sum_{k=1}^{mn} \mathcal{E}_{\pi_0} \left(\sum_{\eta_k \in \Pi_{S_{Q(\theta)}}} \mathcal{E}_{\eta_k} \circ \mathcal{E}_{\pi_1} \left(\cdots \left(\sum_{\eta_1 \in \Pi_{S_{Q(\theta)}}} \mathcal{E}_{\eta_1} \circ \mathcal{E}_{\pi_1}(\rho) \right) \cdots \right) \right) \\
&= \mathcal{E}_{\pi_0}(\rho) + \sum_{k=1}^{mn} \mathcal{E}_{\pi_0} (\llbracket Q(\theta) \rrbracket \circ \mathcal{E}_{\pi_1})^k (\rho) \quad (\text{by inductive hypothesis}) \\
&\sqsubseteq \llbracket \text{while } M[\bar{q}] = 1 \text{ do } Q(\theta) \text{ od} \rrbracket (\rho) \tag{D.6}
\end{aligned}$$

and

$$\begin{aligned}
& \sum_{\pi \in \Pi_{S_P(\theta)}^{((\sum_{j=1}^m n_j) + m + 1)}} \mathcal{E}_\pi(\rho) \\
& \supseteq \mathcal{E}_{\pi_0}(\rho) + \sum_{k=1}^m \sum_{\eta_k \in \Pi_{S_Q(\theta)}^{(n_k)}} \cdots \sum_{\eta_1 \in \Pi_{S_Q(\theta)}^{(n_1)}} \mathcal{E}_{\pi_1 \eta_1 \pi_1 \eta_2 \cdots \pi_1 \eta_k \pi_0}(\rho) \\
& = \mathcal{E}_{\pi_0}(\rho) + \sum_{k=1}^m \mathcal{E}_{\pi_0} \left(\sum_{\eta_k \in \Pi_{S_Q(\theta)}^{(n_k)}} \mathcal{E}_{\eta_k} \circ \mathcal{E}_{\pi_1} \left(\cdots \left(\sum_{\eta_1 \in \Pi_{S_Q(\theta)}^{(n_1)}} \mathcal{E}_{\eta_1} \circ \mathcal{E}_{\pi_1}(\rho) \right) \cdots \right) \right).
\end{aligned}$$

Use the inductive hypothesis and let $n_1 \rightarrow \infty$, we have

$$\begin{aligned}
& \sum_{\pi \in \Pi_{S_P(\theta)}} \mathcal{E}_\pi(\rho) \\
& \supseteq \mathcal{E}_{\pi_0}(\rho) + \sum_{k=1}^m \mathcal{E}_{\pi_0} \left(\sum_{\eta_k \in \Pi_{S_Q(\theta)}^{(n_k)}} \mathcal{E}_{\eta_k} \circ \mathcal{E}_{\pi_1} \left(\cdots \left(\sum_{\eta_2 \in \Pi_{S_Q(\theta)}^{(n_2)}} \mathcal{E}_{\eta_1} \circ \mathcal{E}_{\pi_1}(\llbracket Q(\theta) \rrbracket \circ \mathcal{E}_{\pi_1}(\rho)) \right) \cdots \right) \right)
\end{aligned}$$

as the same, let $n_2 \rightarrow \infty, \dots, n_m \rightarrow \infty$ in order, we get

$$\sum_{\pi \in \Pi_{S_P(\theta)}} \mathcal{E}_\pi(\rho) \supseteq \mathcal{E}_{\pi_0}(\rho) + \sum_{k=1}^m \mathcal{E}_{\pi_0}(\llbracket Q(\theta) \rrbracket \circ \mathcal{E}_{\pi_1})^k(\rho),$$

then let $m \rightarrow \infty$,

$$\sum_{\pi \in \Pi_{S_P(\theta)}} \mathcal{E}_\pi(\rho) \supseteq \llbracket \text{while } M[\bar{q}] = 1 \text{ do } Q(\theta) \text{ od} \rrbracket(\rho).$$

Back to the Formula (D.6), let $m \rightarrow \infty$, we have

$$\sum_{\pi \in \Pi_{S_P(\theta)}} \mathcal{E}_\pi(\rho) \subseteq \llbracket \text{while } M[\bar{q}] = 1 \text{ do } Q(\theta) \text{ od} \rrbracket(\rho).$$

Therefore,

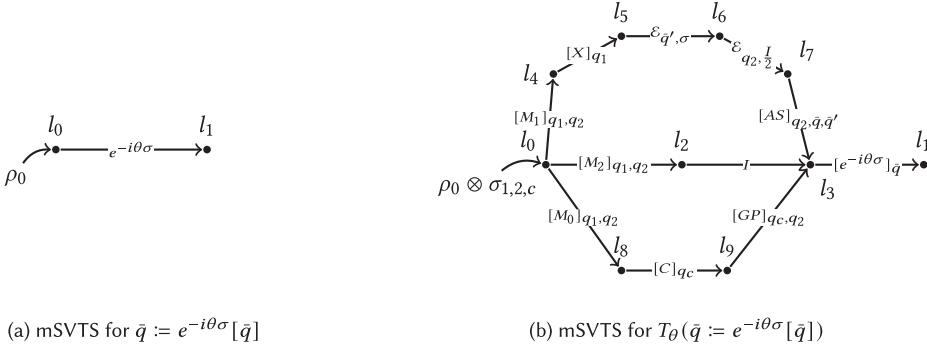
$$\sum_{\pi \in \Pi_{S_P(\theta)}} \mathcal{E}_\pi(\rho) = \llbracket \text{while } M[\bar{q}] = 1 \text{ do } Q(\theta) \text{ od} \rrbracket(\rho).$$

□

Figure 11 shows the mSVTS for $\bar{q} := e^{-i\theta\sigma}[\bar{q}]$ and the mSVTS for $T_\theta(\bar{q} := e^{-i\theta\sigma}[\bar{q}])$. With this, we can construct an mSVTS for $\frac{\partial}{\partial\theta}(P(\theta))$ by only modifying the mSVTS of $P(\theta)$.

LEMMA D.8. Let $S_{P(\theta)}$ be the control-flow graph of a parameterized quantum **while**-program $P(\theta)$. We can modify $S_{P(\theta)}$ to be the control-flow graph of $Q(\theta) \equiv \frac{\partial}{\partial\theta}(P(\theta)) = \text{Dinit}; T_\theta(P(\theta))$ as follows:

- (1) For **Dinit**: We add 4 locations $l_{in}^Q, l_1^Q, l_2^Q, l_3^Q, l_4^Q$ and five transitions $l_{in}^Q \xrightarrow{\mathcal{E}_{q_1}} l_1^Q, l_1^Q \xrightarrow{\mathcal{E}_{q_2}} l_2^Q, l_2^Q \xrightarrow{\mathcal{E}_{q_c}} l_3^Q, l_3^Q \xrightarrow{[C]_{q_c}} l_4^Q, l_4^Q \xrightarrow{[G^P]_{q_c, q_2}} l_{in}^{P(\theta)}$.

Fig. 11. Example of mSVTS for parameterized quantum **while**-program.

- (2) For $T_\theta(P(\theta))$: As shown in Figure 11, for every transition $a \xrightarrow{[e^{-i\theta\sigma}]_{\bar{q}}} b$ with a parameter symbol θ in $\mathcal{S}_{P(\theta)}$, we add 8 locations: $l_{a,b,j}, 2 \leq j \leq 9$ and replace the transition $a \xrightarrow{e^{-i\theta\sigma}} b$ by 11 transitions: $a \xrightarrow{[M_1]_{q_1,q_2}} l_{a,b,4} \xrightarrow{[X]_{q_1}} l_{a,b,5} \xrightarrow{\varepsilon_{q',\sigma}} l_{a,b,6} \xrightarrow{\varepsilon_{q_2, \frac{I}{2}}} l_{a,b,7} \xrightarrow{[AS]_{q_2,\bar{q},q'}} l_{a,b,9} \xrightarrow{[GP]_{q_c,q_2}} l_{a,b,3} \xrightarrow{[e^{-i\theta\sigma}]_{\bar{q}}} b$, $a \xrightarrow{[M_2]_{q_1,q_2}} l_{a,b,2} \xrightarrow{I} l_{a,b,3}$, $a \xrightarrow{[M_0]_{q_1,q_2}} l_{a,b,8} \xrightarrow{[C]_{q_c}} l_{a,b,9}$.

Then, we get an mSVTS $\mathcal{S}_{Q(\theta)}$, we have that $\mathcal{S}_{Q(\theta)}$ represents the control-flow graph of $Q(\theta)$.

PROOF. By definition, the control-flow graph of $Q(\theta) = \text{Dinit}; T_\theta(P(\theta))$ can be constructed from $\mathcal{S}_{\text{Dinit}}$ and $\mathcal{S}_{T_\theta(P(\theta))}$. In (1), it is obvious that (1) constructs the control-flow graph of the program **Dinit**. We next prove that in (2), it produces a control-flow graph for $T_\theta(P(\theta))$. We prove this by induction through the program structure of $P(\theta)$.

- $P(\theta) \equiv \text{skip}$, or $q := |0\rangle$, or $\bar{q} := U[\bar{q}]$, or $\bar{q} := e^{-i\theta'\sigma}[\bar{q}]$ (the symbol $\theta' \neq \theta$). By definition of T_θ , $T_\theta(P(\theta)) = P(\theta)$ and we also see that $P(\theta)$ does not contain statement using θ , then $\mathcal{S}_{P(\theta)}$ has no transition that contains θ . Thus, in (2), we do not change the $\mathcal{S}_{P(\theta)}$. We have that it is still $\mathcal{S}_{P(\theta)}$.
- $P(\theta) \equiv \bar{q} := e^{-i\theta\sigma}[\bar{q}]$. In (2), our construction comes from the Figure 11; we can check that the outcome represents the control-flow graph of $T_\theta(\bar{q} := e^{-i\theta\sigma}[\bar{q}])$.
- $P(\theta) \equiv P_1(\theta); P_2(\theta)$. Using

$$T_\theta(P_1(\theta); P_2(\theta)) = T_\theta(P_1(\theta)); T_\theta(P_2(\theta))$$

and the inductive hypothesis on $P_1(\theta)$ and $P_2(\theta)$, the replacements in (2) are carried internally in $P_1(\theta)$ and $P_2(\theta)$, then concatenate them. This procedure is what we do in the definition of the control-flow graph with mSVTS. Thus, the outcome represents the control-flow graph of $T_\theta(P(\theta))$.

- $P(\theta) \equiv \text{if } (\Box m \cdot M[\bar{q}] = m \rightarrow P_m(\theta)) \text{ fi}$. We have $T_\theta(\text{if } (\Box m \cdot M[\bar{q}] = m \rightarrow P_m(\theta)) \text{ fi}) = \text{if } (\Box m \cdot M[\bar{q}] = m \rightarrow T_\theta(P_m(\theta))) \text{ fi}$ and the inductive hypothesis on $P_m(\theta)$. Then, the rest is as same as above.
- $P(\theta) \equiv \text{while } M[\bar{q}] = 1 \text{ do } P'(\theta) \text{ od}$. We have that $T_\theta(\text{while } M[\bar{q}] = 1 \text{ do } P'(\theta) \text{ od}) = \text{while } M[\bar{q}] = 1 \text{ do } T_\theta(P'(\theta))$ and the inductive hypothesis on $P'(\theta)$. Then, the rest is as same as above.

Therefore, we get $\mathcal{S}_{\text{Dinit}}$ and $\mathcal{S}_{T_\theta(P(\theta))}$. Since $\mathcal{S}_{\text{Dinit}}$ has the exit location $l_{in}^{P(\theta)}$ and $\mathcal{S}_{T_\theta(P(\theta))}$ has the same location $l_{in}^{P(\theta)}$ as the initial location, we have that $\mathcal{S}_{Q(\theta)}$ represents the control-flow graph of $\frac{\partial}{\partial \theta}(P(\theta))$. $\square$

As the same settings in Lemma D.8, let $\eta_{\text{Dinit}} = l_{in}^Q \xrightarrow{\mathcal{E}_{q_1}} l_1^Q \xrightarrow{\mathcal{E}_{q_2}} l_2^Q \xrightarrow{\mathcal{E}_{q_c}} l_3^Q \xrightarrow{[C]_{q_c}} l_4^Q \xrightarrow{[GP]_{q_c, q_2}} l_{in}^{P(\theta)}$ and for every transition $a \xrightarrow{[e^{-i\theta\sigma}]_{\bar{q}}} b$ with parameter symbol θ in $\mathcal{S}_{P(\theta)}$, we write

$$\begin{aligned} \eta_{a,b,0} &= a \xrightarrow{[M_0]_{q_1, q_2}} l_{a,b,8} \xrightarrow{[C]_{q_c}} l_{a,b,9} \xrightarrow{[GP]_{q_c, q_2}} l_{a,b,3} \xrightarrow{[e^{-i\theta\sigma}]_{\bar{q}}} b \\ \eta_{a,b,1} &= a \xrightarrow{[M_1]_{q_1, q_2}} l_{a,b,4} \xrightarrow{[X]_{q_1}} l_{a,b,5} \xrightarrow{\mathcal{E}_{q', \sigma}} l_{a,b,6} \xrightarrow{\mathcal{E}_{q_2, \frac{1}{2}}} l_{a,b,7} \xrightarrow{[AS]_{q_2, q, q'}} l_{a,b,3} \xrightarrow{[e^{-i\theta\sigma}]_{\bar{q}}} b \\ \eta_{a,b,2} &= a \xrightarrow{[M_2]_{q_1, q_2}} l_{a,b,2} \xrightarrow{I} l_{a,b,3} \xrightarrow{[e^{-i\theta\sigma}]_{\bar{q}}} b. \end{aligned}$$

For each $\pi \in \Pi_{\mathcal{S}_{P(\theta)}}$ with $\pi = l_{in}^{P(\theta)} \xrightarrow{\mathcal{E}_1} m_1 \xrightarrow{\mathcal{E}_2} m_2 \xrightarrow{\mathcal{E}_3} \dots \xrightarrow{\mathcal{E}_n} m_n$ and $1 \leq i_1 < i_2 < \dots < i_k \leq n$ such that $\mathcal{E}_{i_j} = [e^{-i\theta\sigma_{i_j}}]_{\bar{q}_{i_j}}$ for $j = 1, 2, \dots, k$ (which is that the path π has k times occurrences of the parameter symbol θ), we write $\pi_j = m_{j-1} \xrightarrow{\mathcal{E}_j} m_j$ for $1 \leq j \leq n$ ($m_0 = l_{in}^{P(\theta)}$) and then define

$$\begin{aligned} A_\pi &= \left\{ \eta_{dinit} \pi_1 \dots \pi_{i_1-1} \mu_{i_1} \pi_{i_1+1} \dots \pi_{i_k-1} \mu_{i_k} \pi_{i_k+1} \dots \pi_n \mid \mu_{i_j} \in \{\eta_{m_{i_j-1}, m_{i_j}, 0}, \eta_{m_{i_j-1}, m_{i_j}, 1}, \eta_{m_{i_j-1}, m_{i_j}, 2}\} \right\}. \end{aligned} \quad (\text{D.7})$$

In there, the set A_π is obtained by replacing each π_{i_j} , which contains parameter symbol θ , with one of $\eta_{m_{i_j-1}, m_{i_j}, 0}$, $\eta_{m_{i_j-1}, m_{i_j}, 1}$, $\eta_{m_{i_j-1}, m_{i_j}, 2}$ and adding η_{dinit} to the front of π . We have the following lemma:

LEMMA D.9. *As the same settings in Lemma D.8, and $A_\pi, \pi \in \Pi_{\mathcal{S}_{P(\theta)}}$ defined above, we have*

$$\Pi_{\mathcal{S}_{Q(\theta)}} = \bigcup_{\pi \in \Pi_{\mathcal{S}_{P(\theta)}}} A_\pi. \quad (\text{D.8})$$

PROOF. By constructions of $\mathcal{S}_{Q(\theta)}$ and A_π , Formula (D.8) is easy to see. $\square$

LEMMA D.10. *As the same settings in Lemma D.8, for any observable O on $\mathcal{H}_{P(\theta)}$, the same O_d in Theorem 5.4 and $\pi \in \Pi_{\mathcal{S}_{P(\theta)}}$ with $\pi = l_{in}^{P(\theta)} \xrightarrow{\mathcal{E}_1} m_1 \xrightarrow{\mathcal{E}_2} m_2 \xrightarrow{\mathcal{E}_3} \dots \xrightarrow{\mathcal{E}_n} m_n$ and $1 \leq i_1 < i_2 < \dots < i_k \leq n$ such that $\mathcal{E}_{i_j} = [e^{-i\theta\sigma_{i_j}}]_{\bar{q}_{i_j}}$ for $j = 1, 2, \dots, k$, we have*

$$\text{tr} \left(O_d \otimes O \sum_{\eta \in A_\pi} \mathcal{E}_\eta(\rho) \right) = \sum_{j=1}^k \left(\text{tr} \left(O \mathcal{E}_{\pi_{i_j} \pi_{i_j+1} \dots \pi_n} \left(-i[[\sigma_{i_j}]_{\bar{q}_{i_j}}, \mathcal{E}_{\pi_1 \dots \pi_{i_j-1}}(\rho)] \right) \right) \right), \quad (\text{D.9})$$

where $[\sigma_{i_j}]_{\bar{q}_{i_j}}$ denotes the operator σ_{i_j} that acts on the Hilbert space $\mathcal{H}_{\bar{q}_{i_j}}$.

PROOF. We only need to consider the state in $\mathcal{H}_{q_c} \otimes \mathcal{H}_{q_1} \otimes \mathcal{H}_{q_2} \otimes \mathcal{H}_{P(\theta)}$.

$$\begin{aligned}
& \sum_{\eta \in A_\pi} \mathcal{E}_\eta(\rho) \\
&= \sum_{\substack{j=1,2,\dots,k \\ \mu_{ij} \in \{\eta_{m_{ij-1}, m_{ij}, 0}, \eta_{m_{ij-1}, m_{ij}, 1}, \eta_{m_{ij-1}, m_{ij}, 2}\}}} \left(\mathcal{E}_{\eta_{dinit} \pi_1 \dots \pi_{i_1-1} \mu_{i_1} \pi_{i_1+1} \dots \pi_{i_k-1} \mu_{i_k} \pi_{i_k+1} \dots \pi_n}(\rho) \right) \\
&= \sum_{\substack{j=1,2,\dots,k \\ \mu_{ij} \in \{\eta_{m_{ij-1}, m_{ij}, 0}, \eta_{m_{ij-1}, m_{ij}, 1}, \eta_{m_{ij-1}, m_{ij}, 2}\}}} \left(\mathcal{E}_{\pi_{i_1+1} \dots \pi_{i_k-1} \mu_{i_k} \pi_{i_k+1} \dots \pi_n} \left(\mathcal{E}_{\eta_{dinit} \pi_1 \dots \pi_{i_1-1} \mu_{i_1}}(\rho) \right) \right) \\
&= \sum_{\substack{j=2,\dots,k \\ \mu_{ij} \in \{\eta_{m_{ij-1}, m_{ij}, 0}, \eta_{m_{ij-1}, m_{ij}, 1}, \eta_{m_{ij-1}, m_{ij}, 2}\}}} \left(\mathcal{E}_{\pi_{i_1+1} \dots \pi_{i_k-1} \mu_{i_k} \pi_{i_k+1} \dots \pi_n} \left(\sum_{h_1=0}^2 \mathcal{E}_{\eta_{m_{i_1-1}, m_{i_1}, h_1}} \left(\mathcal{E}_{\eta_{dinit} \pi_1 \dots \pi_{i_1-1}}(\rho) \right) \right) \right).
\end{aligned}$$

Let $|\psi_j\rangle = R_y(2 \arcsin(\sqrt{b_j}))|0\rangle$, $b_j = \mu(j)/(1 - \sum_{k=1}^{j-1} \mu(k))$, then with Lemma D.3 and definition of $\mathcal{E}_{\eta_{dinit} \pi_1 \dots \pi_{i_1-1}}$, we have

$$\begin{aligned}
& \sum_{h_1=0}^2 \mathcal{E}_{\eta_{m_{i_1-1}, m_{i_1}, h_1}} (\mathcal{E}_{\eta_{dinit} \pi_1 \dots \pi_{i_1-1}}(\rho)) \\
&= \sum_{h_1=0}^2 \mathcal{E}_{\eta_{m_{i_1-1}, m_{i_1}, h_1}} \left(|1\rangle_c \langle 1| \otimes |0\rangle_1 \langle 0| \otimes |\psi_1\rangle_2 \langle \psi_1| \otimes \mathcal{E}_{\pi_1 \dots \pi_{i_1-1}}(\rho) \right) \\
&= \left(\sum_{l=2}^{\infty} \mu(l) \right) |2\rangle_c \langle 2| \otimes |0\rangle_1 \langle 0| \otimes |\psi_2\rangle_2 \langle \psi_2| \otimes \mathcal{E}_{\pi_1 \dots \pi_{i_1}}(\rho) \\
&\quad + \frac{\mu(1)}{2} |1\rangle_c \langle 1| \otimes |1\rangle_1 \langle 1| \\
&\quad \otimes \left(|0\rangle_2 \langle 0| \otimes \left(\cos\left(\frac{\pi}{4}\right)^2 \mathcal{E}_{\pi_1 \dots \pi_{i_1}}(\rho) + \sin\left(\frac{\pi}{4}\right)^2 \mathcal{E}_{\pi_{i_1}}(\text{tr}_{\bar{q}_{i_1}}(\mathcal{E}_{\pi_1 \dots \pi_{i_1-1}}(\rho))) \otimes \sigma_{i_1} \right) \right. \\
&\quad \left. + \frac{1}{2} \sin\left(\frac{\pi}{2}\right) \mathcal{E}_{\pi_{i_1}}(-i[[\sigma_{i_1}]_{\bar{q}_{i_1}}, \mathcal{E}_{\pi_1 \dots \pi_{i_1-1}}(\rho)]) \right) \\
&\quad |1\rangle_2 \langle 1| \otimes \left(\cos\left(-\frac{\pi}{4}\right)^2 \mathcal{E}_{\pi_1 \dots \pi_{i_1}}(\rho) + \sin\left(-\frac{\pi}{4}\right)^2 \mathcal{E}_{\pi_{i_1}}(\text{tr}_{\bar{q}_{i_1}}(\mathcal{E}_{\pi_1 \dots \pi_{i_1-1}}(\rho))) \otimes \sigma_{i_1} \right) \\
&\quad \left. + \frac{1}{2} \sin\left(-\frac{\pi}{2}\right) \mathcal{E}_{\pi_{i_1}}(-i[[\sigma_{i_1}]_{\bar{q}_{i_1}}, \mathcal{E}_{\pi_1 \dots \pi_{i_1-1}}(\rho)]) \right).
\end{aligned}$$

We find that in the second term q_1 is in $|1\rangle\langle 1|$, then in the later execution, it will never go into M_1 or M_2 , thus,

$$\begin{aligned}
& \sum_{h_2=0}^2 \sum_{h_1=0}^2 \mathcal{E}_{\eta_{m_{i_2-1}, m_{i_2}, h_2}} \left(\mathcal{E}_{\pi_{i_1+1} \dots \pi_{i_2-1}} \left(\mathcal{E}_{\eta_{m_{i_1-1}, m_{i_1}, h_1}} \left(\mathcal{E}_{\eta_{dinit} \pi_1 \dots \pi_{i_1-1}}(\rho) \right) \right) \right) \\
&= \left(\sum_{l=3}^{\infty} \mu(l) \right) |3\rangle_c \langle 3| \otimes |0\rangle_1 \langle 0| \otimes |\psi_3\rangle_2 \langle \psi_3| \otimes \mathcal{E}_{\pi_1 \dots \pi_{i_2}}(\rho) \\
&\quad + \frac{\mu(2)}{2} |2\rangle_c \langle 2| \otimes |1\rangle_1 \langle 1| \\
&\quad \otimes \left(|0\rangle_2 \langle 0| \otimes \left(\cos\left(\frac{\pi}{4}\right)^2 \mathcal{E}_{\pi_1 \dots \pi_{i_2}}(\rho) + \sin\left(\frac{\pi}{4}\right)^2 \mathcal{E}_{\pi_{i_2}}(\text{tr}_{\bar{q}_{i_2}}(\mathcal{E}_{\pi_1 \dots \pi_{i_2-1}}(\rho))) \otimes \sigma_{i_2} \right) \right.
\end{aligned}$$

$$\begin{aligned}
& + \frac{1}{2} \sin\left(\frac{\pi}{2}\right) \mathcal{E}_{\pi_{i_2}}(-i[[\sigma_{i_2}]_{\bar{q}_{i_2}}, \mathcal{E}_{\pi_1 \dots \pi_{i_2-1}}(\rho)]) \\
& |1\rangle_2 \langle 1| \otimes \left(\cos\left(-\frac{\pi}{4}\right)^2 \mathcal{E}_{\pi_1 \dots \pi_{i_2}}(\rho) + \sin\left(-\frac{\pi}{4}\right)^2 \mathcal{E}_{\pi_{i_2}}(\text{tr}_{\bar{q}_{i_2}}(\mathcal{E}_{\pi_1 \dots \pi_{i_2-1}}(\rho)) \otimes \sigma_{i_2}) \right. \\
& \left. + \frac{1}{2} \sin\left(-\frac{\pi}{2}\right) \mathcal{E}_{\pi_{i_2}}(-i[[\sigma_{i_2}]_{\bar{q}_{i_2}}, \mathcal{E}_{\pi_1 \dots \pi_{i_2-1}}(\rho)]) \right) \\
& + \frac{\mu(1)}{2} |1\rangle_c \langle 1| \otimes |1\rangle_1 \langle 1| \\
& \otimes \left(|0\rangle_2 \langle 0| \otimes \left(\cos\left(\frac{\pi}{4}\right)^2 \mathcal{E}_{\pi_1 \dots \pi_{i_1}}(\rho) + \sin\left(\frac{\pi}{4}\right)^2 \mathcal{E}_{\pi_{i_1}}(\text{tr}_{\bar{q}_{i_1}}(\mathcal{E}_{\pi_1 \dots \pi_{i_1-1}}(\rho)) \otimes \sigma_{i_1}) \right. \right. \\
& \left. + \frac{1}{2} \sin\left(\frac{\pi}{2}\right) \mathcal{E}_{\pi_{i_1}}(-i[[\sigma_{i_1}]_{\bar{q}_{i_1}}, \mathcal{E}_{\pi_1 \dots \pi_{i_1-1}}(\rho)]) \right) \\
& |1\rangle_2 \langle 1| \otimes \left(\cos\left(-\frac{\pi}{4}\right)^2 \mathcal{E}_{\pi_1 \dots \pi_{i_1}}(\rho) + \sin\left(-\frac{\pi}{4}\right)^2 \mathcal{E}_{\pi_{i_1}}(\text{tr}_{\bar{q}_{i_1}}(\mathcal{E}_{\pi_1 \dots \pi_{i_1-1}}(\rho)) \otimes \sigma_{i_1}) \right. \\
& \left. + \frac{1}{2} \sin\left(-\frac{\pi}{2}\right) \mathcal{E}_{\pi_{i_1}}(-i[[\sigma_{i_1}]_{\bar{q}_{i_1}}, \mathcal{E}_{\pi_1 \dots \pi_{i_1-1}}(\rho)]) \right)
\end{aligned}$$

step-by-step,

$$\begin{aligned}
\sum_{\eta \in A_\pi} \mathcal{E}_\eta(\rho) &= \left(\sum_{l=k+1}^{\infty} \mu(l) \right) |k+1\rangle_c \langle k+1| \otimes |0\rangle_1 \langle 0| \otimes |\psi_{k+1}\rangle_2 \langle \psi_{k+1}| \otimes \mathcal{E}_\pi(\rho) \\
&+ \sum_{j=1}^k \left(\frac{\mu(j)}{2} |j\rangle_c \langle j| \otimes |1\rangle_1 \langle 1| \right. \\
&\otimes \left(|0\rangle_2 \langle 0| \otimes \left(\frac{1}{2} \mathcal{E}_\pi(\rho) + \frac{1}{2} \mathcal{E}_{\pi_{i_j} \pi_{i_j+1} \dots \pi_n}(\text{tr}_{\bar{q}_{i_j}}(\mathcal{E}_{\pi_1 \dots \pi_{i_j-1}}(\rho)) \otimes \sigma_{i_j}) \right. \right. \\
&\left. \left. + \frac{1}{2} \mathcal{E}_{\pi_{i_j} \pi_{i_j+1} \dots \pi_n}(-i[[\sigma_{i_j}]_{\bar{q}_{i_j}}, \mathcal{E}_{\pi_1 \dots \pi_{i_j-1}}(\rho)]) \right) \right) \\
&|1\rangle_2 \langle 1| \otimes \left(\frac{1}{2} \mathcal{E}_\pi(\rho) + \frac{1}{2} \mathcal{E}_{\pi_{i_j} \pi_{i_j+1} \dots \pi_n}(\text{tr}_{\bar{q}_{i_j}}(\mathcal{E}_{\pi_1 \dots \pi_{i_j-1}}(\rho)) \otimes \sigma_{i_j}) \right. \\
&\left. \left. - \frac{1}{2} \mathcal{E}_{\pi_{i_j} \pi_{i_j+1} \dots \pi_n}(-i[[\sigma_{i_j}]_{\bar{q}_{i_j}}, \mathcal{E}_{\pi_1 \dots \pi_{i_j-1}}(\rho)]) \right) \right).
\end{aligned}$$

With $O_d = \sum_{j=1}^{\infty} \frac{2}{\mu(j)} |j\rangle \langle j| \otimes |1\rangle \langle 1| \otimes Z$, an observable on $\mathcal{H}_{q_c} \otimes \mathcal{H}_{q_1} \otimes \mathcal{H}_{q_2}$, we have

$$\text{tr} \left(O_d \otimes O \sum_{\eta \in A_\pi} \mathcal{E}_\eta(\rho) \right) = \sum_{j=1}^k \left(\text{tr} \left(O \mathcal{E}_{\pi_{i_j} \pi_{i_j+1} \dots \pi_n}(-i[[\sigma_{i_j}]_{\bar{q}_{i_j}}, \mathcal{E}_{\pi_1 \dots \pi_{i_j-1}}(\rho)]) \right) \right).$$

□

We already know that

$$\left\| \frac{\partial}{\partial \theta} (P(\theta)) \right\| (\rho) = \sum_{\pi \in \Pi_{S_{Q(\theta)}}} \mathcal{E}_\pi(\rho) = \sum_{\pi \in \Pi_{S_{P(\theta)}}} \sum_{\eta \in A_\pi} \mathcal{E}_\eta(\rho),$$

then,

$$\text{tr} \left(O_d \otimes O \left\| \frac{\partial}{\partial \theta} (P(\theta)) \right\| (\rho) \right) = \sum_{\pi \in \Pi_{S_{P(\theta)}}} \text{tr} \left(O_d \otimes O \sum_{\eta \in A_\pi} \mathcal{E}_\eta(\rho) \right). \quad (\text{D.10})$$

We should carefully consider the convergence of the above summation.

LEMMA D.11. As the same settings in Theorem 5.4 and Lemma D.8, we fix $\theta = \theta^*$, for $x \in \mathbb{R}$, $n \in \mathbb{N}$, let

$$h_n(x) = \sum_{\pi \in \Pi_{S_{P(\theta^*[\theta \mapsto x])}}^{(n)}} \text{tr} \left(O_d \otimes O \sum_{\eta \in A_\pi} \mathcal{E}_\eta(\rho) \right),$$

then, $\lim_{n \rightarrow \infty} h_n(x)$ exists, which is exactly

$$\text{tr} \left(O_d \otimes O \left\| \frac{\partial}{\partial \theta} (P(\theta^*[\theta \mapsto x])) \right\| (\rho) \right)$$

and $h_n(x)$ is uniform convergent on any close interval.

PROOF. Let M be the largest eigenvalue of $|O|$. For any $\pi \in \Pi_{S_{P(\theta)}}$ with $\pi = l_{in}^{P(\theta)} \xrightarrow{\mathcal{E}_1} m_1 \xrightarrow{\mathcal{E}_2} m_2 \xrightarrow{\mathcal{E}_3} \dots \xrightarrow{\mathcal{E}_n} m_n$, if symbol θ does not appear in π , then

$$\left| \text{tr} \left(O_d \otimes O \sum_{\eta \in A_\pi} \mathcal{E}_\eta(\rho) \right) \right| = 0.$$

Otherwise, there exist $1 \leq i_1 < i_2 < \dots < i_k \leq n$ such that $\mathcal{E}_{i_j} = [e^{-i\theta\sigma_{i_j}}]_{\bar{q}_{i_j}}$ for $j = 1, 2, \dots, k$, then by Lemma D.10, we have

$$\text{tr} \left(O_d \otimes O \sum_{\eta \in A_\pi} \mathcal{E}_\eta(\rho) \right) = \sum_{j=1}^k \left(\text{tr} (O \mathcal{E}_{\pi_{i_j} \pi_{i_j+1} \dots \pi_n} (-i[[\sigma_{i_j}]_{\bar{q}_{i_j}}, \mathcal{E}_{\pi_1 \dots \pi_{i_j-1}}(\rho)])) \right),$$

then,

$$\begin{aligned} \left| \text{tr} \left(O_d \otimes O \sum_{\eta \in A_\pi} \mathcal{E}_\eta(\rho) \right) \right| &\leq \sum_{j=1}^k \left| \text{tr} (O \mathcal{E}_{\pi_{i_j} \pi_{i_j+1} \dots \pi_n} (-i[[\sigma_{i_j}]_{\bar{q}_{i_j}}, \mathcal{E}_{\pi_1 \dots \pi_{i_j-1}}(\rho)])) \right| \\ &\leq \sum_{j=1}^k M \left| \text{tr} (\mathcal{E}_{\pi_{i_j} \pi_{i_j+1} \dots \pi_n} (-i[[\sigma_{i_j}]_{\bar{q}_{i_j}}, \mathcal{E}_{\pi_1 \dots \pi_{i_j-1}}(\rho)])) \right| \\ &\leq \sum_{j=1}^k 2M \text{tr} (\mathcal{E}_{\pi_{i_j} \pi_{i_j+1} \dots \pi_n} (\mathcal{E}_{\pi_1 \dots \pi_{i_j-1}}(\rho))) \\ &= \sum_{j=1}^k 2M \text{tr} (\mathcal{E}_\pi(\rho)) = 2kM \text{tr} (\mathcal{E}_\pi(\rho)). \end{aligned}$$

Thus, for any $\pi \in \Pi_{S_{P(\theta)}}$ with length n ,

$$\left| \text{tr} \left(O_d \otimes O \sum_{\eta \in A_\pi} \mathcal{E}_\eta(\rho) \right) \right| \leq 2nM \text{tr} (\mathcal{E}_\pi(\rho)). \quad (\text{D.11})$$

Let M_1 be the size of transition set of $S_{P(\theta)}$, M_2 be the number of occurrences of **while** statements in $P(\theta)$, which means that $P(\theta)$ contains following subprograms:

$$P_j(\theta) \equiv \text{while } M^{(j)}[\bar{q}_j] = 1 \text{ do } Q_j(\theta) \text{ od}, \quad j = 1, \dots, M_2.$$

These $P_j(\theta)$ can be nested within each other. In there, we assume $M_2 \geq 1$, otherwise $P(\theta)$ does not contain **while** statement, then the conclusion is trivial. By Lemma 5.3, we fix θ and choose $\epsilon = \frac{1}{2}$, then for these M_2 subprograms, there exist $N_1, \dots, N_{M_2}$ such that

$$\forall n \geq 0, \forall \rho \in \mathcal{H}_{P(\theta)}, 1 \leq j \leq M_2,$$

$$\text{tr}(\mathcal{E}_0^{(j)} \circ (\llbracket Q_j(\theta) \rrbracket \circ \mathcal{E}_1^{(j)})^n(\rho)) \leq \left(\frac{1}{2}\right)^{\lfloor \frac{n}{N_j} \rfloor} \text{tr}(\rho).$$

Let $N = \max_{j=1, \dots, M_2} N_j$, then

$$\forall n \geq 0, \forall \rho \in \mathcal{H}_{P(\theta)}, 1 \leq j \leq M_2,$$

$$\text{tr}(\mathcal{E}_0^{(j)} \circ (\llbracket Q_j(\theta) \rrbracket \circ \mathcal{E}_1^{(j)})^n(\rho)) \leq \left(\frac{1}{2}\right)^{\lfloor \frac{n}{N} \rfloor} \text{tr}(\rho). \quad (\text{D.12})$$

For any $n \geq 2$, we consider $\pi \in \Pi_{\mathcal{S}_{P(\theta)}}^{((n+1)^{M_2} M_1 - 1)} \setminus \Pi_{\mathcal{S}_{P(\theta)}}^{(n^{M_2} M_1 - 1)}$, the length of π is at least $n^{M_2} M_1$, then π has at least $n^{M_2} M_1 / M_1 = n^{M_2}$ locations in $\mathcal{S}_{P(\theta)}$ repeatedly appear, which is caused by **while** statements. Then, π has at least n^{M_2} times runs into the loop bodies of these $P_j(\theta), j = 1, \dots, M_2$.

Let $a(\{P_1(\theta), \dots, P_{M_2}(\theta)\})$ denote the possible maximum times of π runs into the loop bodies of these $P_j(\theta), j = 1, \dots, M_2$ on the condition that π only continuously runs into the loop body of each $P_j(\theta)$ with no more than $n - 1$ times. We can assume that $P_1(\theta)$ is the first **while** statement to appear in $P(\theta)$ and it contains $0 \leq t \leq M_2 - 1$ **while** statement $P_{j_1}(\theta), \dots, P_{j_t}(\theta), 2 \leq j_1 \leq \dots \leq j_t \leq M_2$, then

$$\begin{aligned} & a(\{P_1(\theta), \dots, P_{M_2}(\theta)\}) \\ & \leq \underbrace{(n-1)}_{\text{for } P_1(\theta)} + \underbrace{(n-1)a(\{P_{j_1}(\theta), \dots, P_{j_t}(\theta)\})}_{\text{for those while statements in the loop body of } P_1(\theta)} \\ & \quad + \underbrace{a(\{P_2(\theta), \dots, P_{M_2}(\theta)\} \setminus \{P_{j_1}(\theta), \dots, P_{j_t}(\theta)\})}_{\text{for those while statements not in the loop body of } P_1(\theta)} \\ & \leq (n-1) + na(\{P_2(\theta), \dots, P_{M_2}(\theta)\}) \\ & \quad \dots \\ & \leq \sum_{j=0}^{M_2-1} (n-1)n^j = n^{M_2} - 1 < n^{M_2}. \end{aligned}$$

This contradicts that π has at least n^{M_2} times runs into the loop body of these $P_j(\theta), j = 1, \dots, M_2$. Therefore, for any $\pi \in \Pi_{\mathcal{S}_{P(\theta)}}^{((n+1)^{M_2} M_1 - 1)} \setminus \Pi_{\mathcal{S}_{P(\theta)}}^{(n^{M_2} M_1 - 1)}$, there exists $1 \leq j_0 \leq M_2$ such that π continuously runs into the loop body of $P_{j_0}(\theta)$ with more than $n - 1$ times, which is π can be written as

$$\pi = \pi_1 \eta_1 \mu_1 \eta_1 \mu_2 \cdots \eta_1 \mu_t \eta_0 \pi_2$$

with $\eta_1 = l_{in}^{P_{j_0}(\theta)} \xrightarrow{\mathcal{E}_1^{(j_0)}} l_{in}^{Q_{j_0}(\theta)}, \eta_0 = l_{in}^{P_{j_0}(\theta)} \xrightarrow{\mathcal{E}_0^{(j_0)}} l_{out}^{P_{j_0}(\theta)}, \mu_j \in \Pi_{\mathcal{S}_{Q_{j_0}(\theta)}}, 1 \leq j \leq t, t \geq n$.

We define the following set for each $1 \leq j \leq M_2, n \geq 2$ and $0 \leq m \leq (n+1)^{M_2} M_1 - 1$:

$$\begin{aligned} A_j^{(n, m)} \equiv & \left\{ \pi \in \Pi_{\mathcal{S}_{P(\theta)}}^{((n+1)^{M_2} M_1 - 1)} \setminus \Pi_{\mathcal{S}_{P(\theta)}}^{(n^{M_2} M_1 - 1)} : \pi = \pi_1 \eta_1 \mu_1 \eta_1 \mu_2 \cdots \eta_1 \mu_n \eta_0 \pi_2, \right. \\ & \eta_1 = l_{in}^{P_j(\theta)} \xrightarrow{\mathcal{E}_1^{(j)}} l_{in}^{Q_j(\theta)}, \eta_0 = l_{in}^{P_j(\theta)} \xrightarrow{\mathcal{E}_0^{(j)}} l_{out}^{P_j(\theta)}, \mu_k \in \Pi_{\mathcal{S}_{Q_j(\theta)}}, 1 \leq k \leq n, \\ & \left. \pi_1 \text{ contains } m \text{ times } \eta_1, \pi_2 \text{ may contain } \eta_1 \right\}, \end{aligned}$$

then,

$$\Pi_{S_{P(\theta)}}^{((n+1)^{M_2} M_1 - 1)} \setminus \Pi_{S_{P(\theta)}}^{(n^{M_2} M_1 - 1)} = \bigcup_{j=1}^{M_2} \left(\bigcup_{m=0}^{n^{M_2} M_1 - 1} A_j^{(n,m)} \right).$$

For each $A_j^{(n,m)}$, we define

$$\begin{aligned} B_j^{(n,m)} &\equiv \left\{ \pi_1 \eta_1 : \pi_1 \eta_1 \mu_1 \eta_1 \mu_2 \cdots \eta_1 \mu_n \eta_0 \pi_2 \in A_j^{(n,m)}, \right. \\ &\quad \eta_1 = l_{in}^{P_j(\theta)} \xrightarrow{\mathcal{E}_1^{(j)}} l_{in}^{Q_j(\theta)}, \eta_0 = l_{in}^{P_j(\theta)} \xrightarrow{\mathcal{E}_0^{(j)}} l_{out}^{P_j(\theta)}, \mu_k \in \Pi_{S_{Q_j(\theta)}}, 1 \leq k \leq n, \\ &\quad \left. \pi_1 \text{ contains } m \text{ times } \eta_1, \pi_2 \text{ may contain } \eta_1 \right\}, \\ C_j^{(n,m)} &\equiv \left\{ \mu_1, \mu_2, \dots, \mu_n : \pi_1 \eta_1 \mu_1 \eta_1 \mu_2 \cdots \eta_1 \mu_n \eta_0 \pi_2 \in A_j^{(n,m)}, \right. \\ &\quad \eta_1 = l_{in}^{P_j(\theta)} \xrightarrow{\mathcal{E}_1^{(j)}} l_{in}^{Q_j(\theta)}, \eta_0 = l_{in}^{P_j(\theta)} \xrightarrow{\mathcal{E}_0^{(j)}} l_{out}^{P_j(\theta)}, \mu_k \in \Pi_{S_{Q_j(\theta)}}, 1 \leq k \leq n, \\ &\quad \left. \pi_1 \text{ contains } m \text{ times } \eta_1, \pi_2 \text{ may contain } \eta_1 \right\}, \\ D_j^{(n,m)} &\equiv \left\{ \pi_2 : \pi_1 \eta_1 \mu_1 \eta_1 \mu_2 \cdots \eta_1 \mu_n \eta_0 \pi_2 \in A_j^{(n,m)}, \right. \\ &\quad \eta_1 = l_{in}^{P_j(\theta)} \xrightarrow{\mathcal{E}_1^{(j)}} l_{in}^{Q_j(\theta)}, \eta_0 = l_{in}^{P_j(\theta)} \xrightarrow{\mathcal{E}_0^{(j)}} l_{out}^{P_j(\theta)}, \mu_k \in \Pi_{S_{Q_j(\theta)}}, 1 \leq k \leq n, \\ &\quad \left. \pi_1 \text{ contains } m \text{ times } \eta_1, \pi_2 \text{ may contain } \eta_1 \right\}. \end{aligned}$$

$B_i^{(n,m)}, C_i^{(n,m)}, D_i^{(n,m)}$ are all finite, and we have,

$$\begin{aligned} A_j^{(n,m)} \subseteq E_i^{(n,m)} &\equiv \left\{ \pi_1 \eta_1 \mu_1 \eta_1 \mu_2 \cdots \eta_1 \mu_n \eta_0 \pi_2 : \pi_1 \eta_1 \in B_i^{(n,m)}, \pi_2 \in D_i^{(n,m)}, \mu_k \in C_i^{(n,m)}, 1 \leq k \leq n \right\}, \\ C_j^{(n,m)} &\subseteq \Pi_{S_{Q_j(\theta)}}. \end{aligned}$$

By Theorem D.7, for any $\rho \in \mathcal{D}(\mathcal{H}_{P(\theta)}), 1 \leq j \leq M_2$,

$$\sum_{\mu \in C_j^{(n,m)}} \mathcal{E}_\mu(\rho) \subseteq \sum_{\mu \in \Pi_{S_{Q_j(\theta)}}} \mathcal{E}_\mu(\rho) = \llbracket Q_j(\theta) \rrbracket(\rho). \quad (\text{D.13})$$

As regards to $B_j^{(n,m)}$, any $\pi, \pi' \in B_j^{(n,m)}, \pi \neq \pi'$, we have $\pi = \pi_1 \eta_1, \pi' = \pi'_1 \eta_1$ and π_1, π'_1 contains m times of η_1 , then π and π' are not prefixes to each other (otherwise $\pi = \pi'$). Then, $B_j^{(n,m)}$ satisfies the condition of Lemma D.5, so we have that for any $\rho \in \mathcal{D}(\mathcal{H}_{P(\theta)}),$

$$\sum_{\pi \in B_j^{(n,m)}} \text{tr}(\mathcal{E}_\pi(\rho)) \leq \text{tr}(\rho), \quad 1 \leq j \leq M_2. \quad (\text{D.14})$$

For any $\pi, \pi' \in D_j^{(n,m)}, \pi \neq \pi', \pi$, and π' have $l_{out}^{P(\theta)}$ as last location, which has no post-location, then π, π' are not prefix of each other. $D_j^{(n,m)}$ also satisfies the conditions of Lemma D.5, then for

any $\rho \in \mathcal{D}(\mathcal{H}_P(\theta))$,

$$\sum_{\pi \in D_j^{(n,m)}} \text{tr}(\mathcal{E}_\pi(\rho)) \leq \text{tr}(\rho), \quad 1 \leq j \leq M_2. \quad (\text{D.15})$$

With Formula (D.13), for any $\rho \in \mathcal{D}(\mathcal{H}_P(\theta))$ and any $1 \leq j \leq M_2$, we have

$$\begin{aligned} & \sum_{\pi \in A_j^{(n,m)}} \mathcal{E}_\pi(\rho) \sqsubseteq \sum_{\pi \in E_i^{(n,m)}} \mathcal{E}_\pi(\rho) \\ &= \sum_{\pi_1 \eta_1 \mu_1 \eta_1 \mu_2 \cdots \eta_1 \mu_n \eta_0 \pi_2 \in E_i^{(n,m)}} \left(\mathcal{E}_{\pi_1 \eta_1 \mu_1 \eta_1 \mu_2 \cdots \eta_1 \mu_n \eta_0 \pi_2}(\rho) \right) \\ &= \sum_{\pi_1 \eta_1 \mu_1 \eta_1 \mu_2 \cdots \eta_1 \mu_n \eta_0 \pi_2 \in E_i^{(n,m)}} \left(\mathcal{E}_{\pi_2} \circ \mathcal{E}_{\eta_0} \circ \mathcal{E}_{\mu_n} \circ \mathcal{E}_{\eta_1} \circ \cdots \circ \mathcal{E}_{\mu_1} \circ \mathcal{E}_{\eta_1} \circ \mathcal{E}_{\pi_1}(\rho) \right) \\ &= \left(\sum_{\pi_2 \in D_j^{(n,m)}} \mathcal{E}_{\pi_2} \right) \circ \mathcal{E}_{\eta_0} \circ \left(\sum_{\mu_n \in C_j^{(n,m)}} \mathcal{E}_{\mu_n} \right) \circ \mathcal{E}_{\eta_1} \circ \cdots \circ \left(\sum_{\mu_1 \in C_j^{(n,m)}} \mathcal{E}_{\mu_1} \right) \circ \left(\sum_{\pi_1 \eta_1 \in B_j^{(n,m)}} \mathcal{E}_{\pi_1 \eta_1} \right)(\rho) \\ &\sqsubseteq \left(\sum_{\pi_2 \in D_j^{(n,m)}} \mathcal{E}_{\pi_2} \right) \circ \mathcal{E}_{\eta_0} \circ \underbrace{\left(\llbracket Q_j(\theta) \rrbracket \circ \mathcal{E}_{\eta_1} \circ \cdots \circ \llbracket Q_j(\theta) \rrbracket \right)}_{n \text{ times } \llbracket Q_j(\theta) \rrbracket} \circ \left(\sum_{\pi_1 \eta_1 \in B_j^{(n,m)}} \mathcal{E}_{\pi_1 \eta_1} \right)(\rho) \\ &= \left(\sum_{\pi_2 \in D_j^{(n,m)}} \mathcal{E}_{\pi_2} \right) \circ \mathcal{E}_{\eta_0} \circ \left(\llbracket Q_j(\theta) \rrbracket \circ \mathcal{E}_{\eta_1} \right)^{n-1} \circ \llbracket Q_j(\theta) \rrbracket \circ \left(\sum_{\pi_1 \eta_1 \in B_j^{(n,m)}} \mathcal{E}_{\pi_1 \eta_1} \right)(\rho), \end{aligned}$$

thus,

$$\begin{aligned} & \sum_{\pi \in A_j^{(n,m)}} \text{tr}(\mathcal{E}_\pi(\rho)) \\ &\leq \text{tr} \left(\left(\sum_{\pi_2 \in D_j^{(n,m)}} \mathcal{E}_{\pi_2} \right) \circ \mathcal{E}_{\eta_0} \circ \left(\llbracket Q_j(\theta) \rrbracket \circ \mathcal{E}_{\eta_1} \right)^{n-1} \circ \llbracket Q_j(\theta) \rrbracket \circ \left(\sum_{\pi_1 \eta_1 \in B_j^{(n,m)}} \mathcal{E}_{\pi_1 \eta_1} \right)(\rho) \right) \\ &\leq \text{tr} \left(\mathcal{E}_{\eta_0} \circ \left(\llbracket Q_j(\theta) \rrbracket \circ \mathcal{E}_{\eta_1} \right)^{n-1} \circ \llbracket Q_j(\theta) \rrbracket \circ \left(\sum_{\pi_1 \eta_1 \in B_j^{(n,m)}} \mathcal{E}_{\pi_1 \eta_1} \right)(\rho) \right) \quad (\text{by Formula (D.15)}) \\ &\leq \left(\frac{1}{2} \right)^{\lfloor \frac{n-1}{N} \rfloor} \text{tr} \left(\llbracket Q_j(\theta) \rrbracket \circ \left(\sum_{\pi_1 \eta_1 \in B_j^{(n,m)}} \mathcal{E}_{\pi_1 \eta_1} \right)(\rho) \right) \quad (\text{by Formula (D.12)}) \\ &\leq \left(\frac{1}{2} \right)^{\lfloor \frac{n-1}{N} \rfloor} \text{tr} \left(\sum_{\pi_1 \eta_1 \in B_j^{(n,m)}} \mathcal{E}_{\pi_1 \eta_1}(\rho) \right) \\ &\leq \left(\frac{1}{2} \right)^{\lfloor \frac{n-1}{N} \rfloor} \text{tr}(\rho). \quad (\text{by Formula (D.14)}) \end{aligned}$$

Then,

$$\begin{aligned} \sum_{\pi \in \Pi_{S_P(\theta)}^{((n+1)^{M_2} M_1 - 1) \setminus \Pi_{S_P(\theta)}^{(n^{M_2} M_1 - 1)}}} \text{tr}(\mathcal{E}_\pi(\rho)) &\leq \sum_{j=1}^{M_2} \sum_{m=0}^{(n+1)^{M_2} M_1 - 1} \sum_{\pi \in A_j^{(n, m)}} \text{tr}(\mathcal{E}_\pi(\rho)) \\ &\leq M_1 M_2 (n+1)^{M_2} \left(\frac{1}{2}\right)^{\lfloor \frac{n-1}{N} \rfloor} \text{tr}(\rho). \end{aligned} \quad (\text{D.16})$$

With Formula (D.11), we have for any $\rho \in \mathcal{D}(\mathcal{H}_{P(\theta)})$,

$$\begin{aligned} &\sum_{\pi \in \Pi_{S_P(\theta)}^{((n+1)^{M_2} M_1 - 1) \setminus \Pi_{S_P(\theta)}^{(n^{M_2} M_1 - 1)}}} \left| \text{tr} \left(O_d \otimes O \sum_{\eta \in A_\pi} \mathcal{E}_\eta(\rho) \right) \right| \\ &\leq \sum_{\pi \in \Pi_{S_P(\theta)}^{((n+1)^{M_2} M_1 - 1) \setminus \Pi_{S_P(\theta)}^{(n^{M_2} M_1 - 1)}}} 2 \left((n+1)^{M_2} M_1 - 1 \right) M \text{tr}(\mathcal{E}_\pi(\rho)) \\ &\leq 2 \left((n+1)^{M_2} M_1 - 1 \right) M M_1 M_2 (n+1)^{M_2} \left(\frac{1}{2}\right)^{\lfloor \frac{n-1}{N} \rfloor} \text{tr}(\rho) \\ &\leq 4 \left((n+1)^{M_2} M_1 - 1 \right) M M_1 M_2 (n+1)^{M_2} \left(\frac{1}{2}\right)^{\frac{n-1}{N}} \text{tr}(\rho). \end{aligned}$$

As the N is dependent on θ , we can obtain that for any θ , there exists $N_\theta > 0$ such that for any $\rho \in \mathcal{D}(\mathcal{H}_{P(\theta)})$ and $n \geq 2$,

$$\begin{aligned} &\sum_{\pi \in \Pi_{S_P(\theta)}^{((n+1)^{M_2} M_1 - 1) \setminus \Pi_{S_P(\theta)}^{(n^{M_2} M_1 - 1)}}} \left| \text{tr} \left(O_d \otimes O \sum_{\eta \in A_\pi} \mathcal{E}_\eta(\rho) \right) \right| \\ &\leq 4 \left((n+1)^{M_2} M_1 - 1 \right) M M_1 M_2 (n+1)^{M_2} \left(\frac{1}{2}\right)^{\frac{n-1}{N_\theta}} \text{tr}(\rho), \end{aligned}$$

then, it is easy to see that for any x , there exists $N_x > 0$ such that for any $\rho \in \mathcal{D}(\mathcal{H}_{P(\theta)})$ and $n \geq 2$,

$$\begin{aligned} H_n(x) &\equiv \sum_{\pi \in \Pi_{S_P(\theta^*[\theta \rightarrow x])}^{((n+1)^{M_2} M_1 - 1) \setminus \Pi_{S_P(\theta^*[\theta \rightarrow x])}^{(n^{M_2} M_1 - 1)}}} \left| \text{tr} \left(O_d \otimes O \sum_{\eta \in A_\pi} \mathcal{E}_\eta(\rho) \right) \right| \\ &\leq 4 \left((n+1)^{M_2} M_1 - 1 \right) M M_1 M_2 (n+1)^{M_2} \left(\frac{1}{2}\right)^{\frac{n-1}{N_x}} \text{tr}(\rho). \end{aligned}$$

With

$$\begin{aligned} \lim_{n \rightarrow \infty} \sqrt[n]{4 \left((n+1)^{M_2} M_1 - 1 \right) M M_1 M_2 (n+1)^{M_2} \left(\frac{1}{2}\right)^{\frac{n-1}{N_x}}} &= \left(\frac{1}{2}\right)^{\frac{1}{N_x}} \\ &< 1, \end{aligned}$$

we have that for any $x \in \mathbb{R}$, $\sum_{n=1}^{\infty} H_n(x)$ is convergent. Since each term of $H_n(x)$ is non-negative, $H_n(x), n \in \mathbb{N}$ is a monotone sequence of continuous functions, then by Dini's Theorem [58, Theorem 7.13], $H_n(x)$ is uniform convergent on any close interval.

Now, for any $n \geq 3$ and any $x \in \mathbb{R}$, we have

$$\begin{aligned}
 |h_{n^{M_2 M_1 - 1}}(x)| &= \left| \sum_{\pi \in \Pi_{S_{P(\theta^*[\theta \mapsto x])}}^{(n^{M_2 M_1 - 1})}} \text{tr} \left(O_d \otimes O \sum_{\eta \in A_\pi} \mathcal{E}_\eta(\rho) \right) \right| \\
 &\leq \sum_{\pi \in \Pi_{S_{P(\theta^*[\theta \mapsto x])}}^{(n^{M_2 M_1 - 1})}} \left| \text{tr} \left(O_d \otimes O \sum_{\eta \in A_\pi} \mathcal{E}_\eta(\rho) \right) \right| \\
 &= \sum_{\pi \in \Pi_{S_{P(\theta^*[\theta \mapsto x])}}^{(2^{M_2 M_1 - 1})}} \left| \text{tr} \left(O_d \otimes O \sum_{\eta \in A_\pi} \mathcal{E}_\eta(\rho) \right) \right| \\
 &\quad + \sum_{k=2}^{n-1} \sum_{\pi \in \Pi_{S_{P(\theta^*[\theta \mapsto x])}}^{((k+1)^{M_2 M_1 - 1})} \setminus \Pi_{S_{P(\theta^*[\theta \mapsto x])}}^{(k^{M_2 M_1 - 1})}} \left| \text{tr} \left(O_d \otimes O \sum_{\eta \in A_\pi} \mathcal{E}_\eta(\rho) \right) \right| \\
 &= \sum_{\pi \in \Pi_{S_{P(\theta^*[\theta \mapsto x])}}^{(2^{M_2 M_1 - 1})}} \left| \text{tr} \left(O_d \otimes O \sum_{\eta \in A_\pi} \mathcal{E}_\eta(\rho) \right) \right| + \sum_{k=2}^{n-1} H_k(x).
 \end{aligned}$$

Because $H_n(x)$ is uniform convergent on any close interval, then $h_n(x)$ is uniform convergent on any close interval. We can also check that

$$\begin{aligned}
 \lim_{n \rightarrow \infty} h_n(x) &= \lim_{n \rightarrow \infty} \sum_{\pi \in \Pi_{S_{P(\theta^*[\theta \mapsto x])}}^{(n)}} \text{tr} \left(O_d \otimes O \sum_{\eta \in A_\pi} \mathcal{E}_\eta(\rho) \right) \\
 &= \sum_{\pi \in \Pi_{S_{P(\theta^*[\theta \mapsto x])}}} \text{tr} \left(O_d \otimes O \sum_{\eta \in A_\pi} \mathcal{E}_\eta(\rho) \right) \\
 &= \text{tr} \left(O_d \otimes O \left[\frac{\partial}{\partial \theta} (P(\theta^*[\theta \mapsto x])) \right] (\rho) \right). \quad (\text{by Equation (D.10)})
 \end{aligned}$$

□

PROOF OF THEOREM 5.4. As the same settings in Lemma D.8, for each $\pi \in \Pi_{P(\theta)}$, we define

$$\begin{aligned}
 f_\pi(\theta) &= \text{tr}(O \mathcal{E}_\pi(\rho)) \\
 g_\pi(\theta) &= \sum_{\eta \in A_\pi} \text{tr}(O_d \otimes O \mathcal{E}_\eta(\rho)).
 \end{aligned}$$

With Lemma D.10, we assume $\pi = l_{in}^{P(\theta)} \xrightarrow{\mathcal{E}_1} m_1 \xrightarrow{\mathcal{E}_2} m_2 \xrightarrow{\mathcal{E}_3} \dots \xrightarrow{\mathcal{E}_n} m_n$ and $1 \leq i_1 < i_2 < \dots < i_k \leq n$ such that $\mathcal{E}_{i_j} = [e^{-i\theta\sigma_{i_j}}]_{q_{i_j}} \text{ for } j = 1, 2, \dots, k$, then

$$g_\pi(\theta) = \sum_{j=1}^k \left(\text{tr} \left(O \mathcal{E}_{\pi_{i_j} \dots \pi_n} (-i[[\sigma_{i_j}]_{q_{i_j}}, \mathcal{E}_{\pi_1 \dots \pi_{i_j-1}}(\rho)]) \right) \right).$$

With Lemma 5.1, we have

$$\frac{\partial}{\partial \theta_j}(f_\pi(\theta)) = \text{tr}\left(O\mathcal{E}_{\pi_{i_j}\dots\pi_n}(-i[[\sigma_{i_j}]_{\hat{q}_{i_j}}, \mathcal{E}_{\pi_1\dots\pi_{i_j-1}}(\rho)])\right)$$

for $j = 1, 2, \dots, k$ (which is considered as the j th occurrence of θ), then

$$\frac{\partial}{\partial \theta}(f_\pi(\theta)) = \sum_{j=1}^k \text{tr}\left(O\mathcal{E}_{\pi_{i_j}\dots\pi_n}(-i[[\sigma_{i_j}]_{\hat{q}_{i_j}}, \mathcal{E}_{\pi_1\dots\pi_{i_j-1}}(\rho)])\right).$$

Thus,

$$\frac{\partial}{\partial \theta}(f_\pi(\theta)) = g_\pi(\theta). \quad (\text{D.17})$$

For $n \geq 1$, let

$$f_n(\theta) = \sum_{\pi \in \Pi_{S_P(\theta)}^{(n)}} f_\pi(\theta)$$

$$g_n(\theta) = \sum_{\pi \in \Pi_{S_P(\theta)}^{(n)}} g_\pi(\theta)$$

we have

$$\lim_{n \rightarrow \infty} f_n(\theta) = f(\theta)$$

$$\lim_{n \rightarrow \infty} g_n(\theta) = g(\theta).$$

The correctness and existence of the second equation are guaranteed by Lemma D.11. By Equation (D.17), we can easily check that for any $n \geq 1$,

$$\frac{\partial}{\partial \theta} f_n(\theta) = g_n(\theta).$$

With Lemma D.11, $g_n(\theta)$ is uniform convergent on a close interval $[\theta - \epsilon, \theta + \epsilon]$ for any $\alpha \in \mathbb{R}$ and any $\epsilon > 0$, which means $\frac{\partial}{\partial \theta}(f_n(\theta))$ is uniform convergent on $[\theta - \epsilon, \theta + \epsilon]$, then

$$\lim_{n \rightarrow \infty} \frac{\partial}{\partial \theta} f_n(\theta) = \frac{\partial}{\partial \theta} \left(\lim_{n \rightarrow \infty} f_n(\theta) \right).$$

Thus,

$$\frac{\partial}{\partial \theta} f(\theta) = \frac{\partial}{\partial \theta} \left(\lim_{n \rightarrow \infty} f_n(\theta) \right) = \lim_{n \rightarrow \infty} \frac{\partial}{\partial \theta} f_n(\theta) = \lim_{n \rightarrow \infty} g_n(\theta) = g(\theta).$$

□

D.5 Proof of Theorem A.3

The proof is based on Appendix D.4. To obtain a more accurate estimation, we define the set $\Gamma_\theta^{(n)} \subseteq \Pi_{S_P(\theta)}$ for a given parameter symbol θ and $n \in \mathbb{N}$:

$$\Gamma_\theta^{(n)} \equiv \{\pi \in \Pi_{S_P(\theta)} : \theta \text{ appears } k \text{ times on path } \pi \text{ and } 0 \leq k \leq n\}.$$

For the set $\Gamma_\theta^{(n)}$, we have a lemma similar to Lemma D.10.

LEMMA D.12. For any $\pi \in \Gamma_\theta^{(n)} \setminus \Gamma_\theta^{(0)}$, $n \geq 1$, π can be written as $\pi = l_{in}^{P(\theta)} \xrightarrow{\mathcal{E}_1} l_1 \xrightarrow{\mathcal{E}_2} l_2 \xrightarrow{\mathcal{E}_3} \dots \xrightarrow{\mathcal{E}_m} l_m$ with $1 \leq i_1 < \dots < i_k \leq m$, $1 \leq k \leq n$ such that $\mathcal{E}_{i_j} = [e^{-i\theta\sigma_{i_j}}]_{q_{i_j}}$ for $j = 1, \dots, k$. These $\mathcal{E}_{i_j}$, $1 \leq j \leq k$ correspond to k times occurrences of θ . Then, for any observable O , we have

$$\text{tr}\left(O_d^2 \otimes O_c^2 \sum_{\eta \in A_\pi} \mathcal{E}_\eta(\rho)\right) = \sum_{j=1}^k \left(\frac{2}{\mu(j)} \left(\text{tr}(O^2 \mathcal{E}_\pi(\rho)) + \text{tr}(O^2 \mathcal{E}_{\pi_{i_j} \pi_{i_{j+1}} \dots \pi_n}(\text{tr}_{q_{i_j}}(\mathcal{E}_{\pi_1 \dots \pi_{i_{j-1}}(\rho)) \otimes \sigma_{i_j})) \right) \right).$$

PROOF. The proof is similar to the proof of Lemma D.10. $\square$

We also need another lemma that is similar to Lemma 5.3.

LEMMA D.13. Consider a quantum loop $P \equiv \text{while } M[\bar{q}] = 1 \text{ do } Q \text{ od}$ with fixed parameters (omitted). Assume that the state space $\mathcal{H}_P$ is finite-dimensional and P terminates almost surely. We define superoperators $\mathcal{E}_i : \mathcal{D}(\mathcal{H}_P) \rightarrow \mathcal{D}(\mathcal{H}_P)$ by $\mathcal{E}_i(\rho) = M_i \rho M_i^\dagger$, $i = 0, 1$ and $\mathcal{E} : \mathcal{D}(\mathcal{H}_P) \rightarrow \mathcal{D}(\mathcal{H}_P)$ by $\mathcal{E}(\rho) = \llbracket Q \rrbracket(\rho)$. Then, for any $\epsilon \in (0, 1)$, there exists $N = N_\epsilon > 0$ such that $\forall n \in \mathbb{N}, \forall \rho \in \mathcal{D}(\mathcal{H}_P)$,

$$\text{tr}((\mathcal{E} \circ \mathcal{E}_1)^n(\rho)) \leq \epsilon^{\lfloor \frac{n}{N} \rfloor} \text{tr}(\rho).$$

PROOF. Because P terminates almost surely, we have that the operator $P_{\mathcal{Y}}$ in Appendix D.1 is an identify operator $P_{\mathcal{Y}} = I$. By Lemma D.2, for any $\epsilon \in (0, 1)$, there exists $N > 0$ such that $(\mathcal{G}^*)^N(P_{\mathcal{Y}}) \sqsubseteq \epsilon P_{\mathcal{Y}}$, which is

$$(\mathcal{G}^*)^N(I) \sqsubseteq \epsilon I.$$

Therefore, for any $n \in \mathbb{N}$, any $\rho \in \mathcal{D}(\mathcal{H}_P)$, we have

$$\text{tr}((\mathcal{E} \circ \mathcal{E}_1)^n(\rho)) = \text{tr}((\mathcal{G}^*)^n(I)\rho) \leq \text{tr}(\epsilon^{\lfloor \frac{n}{N} \rfloor} I \rho) = \epsilon^{\lfloor \frac{n}{N} \rfloor} \text{tr}(\rho).$$

$\square$

We then follow the previous proof of Lemma D.11.

PROOF OF THEOREM A.3. With the Lemma D.12 and a similar proof of Inequality (D.11), we have that for any $\pi \in \Gamma_\theta^{(n)}$,

$$\begin{aligned} \left| \text{tr}\left(O^2 \otimes O_c^2 \sum_{\eta \in A_\pi} \mathcal{E}_\eta(\rho)\right) \right| &\leq \sum_{j=1}^{k_\pi} \left(\frac{2}{\mu(j)} \left(M^2 \text{tr}(\mathcal{E}_\pi(\rho)) + M^2 \text{tr}(\mathcal{E}_{\pi_{i_j} \pi_{i_{j+1}} \dots \pi_n}(\sigma_{\pi_{i_j}})) \right) \right) \\ &= \sum_{j=1}^{k_\pi} \left(\frac{2M^2}{\mu(j)} \text{tr}(\mathcal{E}_\pi(\rho)) + \frac{2M^2}{\mu(j)} \text{tr}(\mathcal{E}_{\pi_{i_j} \pi_{i_{j+1}} \dots \pi_n}(\sigma_{\pi_{i_j}})) \right), \end{aligned} \quad (\text{D.18})$$

where M is the largest eigenvalue of $|O|$, k_π is the k in Lemma D.12 and

$$\sigma_{\pi_{i_j}} \equiv \text{tr}_{q_{i_j}}(\mathcal{E}_{\pi_1 \dots \pi_{i_{j-1}}(\rho)) \otimes \sigma_{i_j}.$$

Let $M_2 = LC(P(\theta))$. For convenience, we assume $M_2 \geq 1$ temporarily. According to the definition of LC (in Definite A.2), $P(\theta)$ contains M_2 subprograms of **while** statements:

$$P_j(\theta) \equiv \text{while } M^{(j)}[\bar{q}_j] = 1 \text{ do } Q_j(\theta) \text{ od}, \quad j = 1, \dots, M_2.$$

With Lemma 5.3, for any $\epsilon \in (0, 1)$, there exists $N_\epsilon \geq 1$ that satisfies the following formula similar to Formula (D.12):

$$\begin{aligned} \forall n \geq 0, \forall \rho \in \mathcal{H}_{P(\theta)}, 1 \leq j \leq M_2, \\ \text{tr}(\mathcal{E}_0^{(j)} \circ (\llbracket Q_j(\theta) \rrbracket \circ \mathcal{E}_1^{(j)})^n(\rho)) \leq \epsilon^{\lfloor \frac{n}{N_\epsilon} \rfloor} \text{tr}(\rho). \end{aligned} \quad (\text{D.19})$$

Let $M_1 = RC_\theta(P(\theta)) \geq 1$. For any $n \geq 1$, we consider $\pi \in \Gamma_\theta^{((n+1)^{M_2} M_1 - 1)} \setminus \Gamma_\theta^{(n^{M_2} M_1 - 1)}$, the parameter symbol θ appears on π at least $n^{M_2} M_1$ times, then π has at least $n^{M_2} M_1 / M_1 = n^{M_2}$ locations of parameter symbol θ repeatedly appear, which is caused by **while** statements. Then, π has at least n^{M_2} times running into the loop bodies of above $P_j(\theta)$, $j = 1, \dots, M_2$. By the same discussion in the proof of Lemma D.11, there exists $1 \leq j_0 \leq M_2$ such that π continuously runs into the loop body of $P_{j_0}(\theta)$ with more than $n - 1$ times.

With the help of Formula (D.19) and Lemma D.5, we can also obtain an inequality similar to Inequality (D.16) in the same way of the proof of Lemma D.11 (note: we can get a tighter bound by limiting $0 \leq m \leq l(k_j)$ for every $1 \leq j \leq M_2$, where k_j is the depth of subprogram $P(\theta)_j$ nested with other $M_2 - 1$ subprograms mentioned before and when $k \geq 2$, $l(k) = (n - 1)^2 n^{k-2}$, $l(1) = 1$),¹⁰

$$\begin{aligned}
 \sum_{\pi \in \Gamma_\theta^{((n+1)^{M_2} M_1 - 1)} \setminus \Gamma_\theta^{(n^{M_2} M_1 - 1)}} \text{tr}(\mathcal{E}_\pi(\rho)) &\leq \sum_{j=1}^{M_2} (l(k_j) + 1) \epsilon^{\lfloor \frac{n-1}{N_\epsilon} \rfloor} \text{tr}(\rho) \\
 &\leq \sum_{j=1}^{M_2} (l(j) + 1) \epsilon^{\lfloor \frac{n-1}{N_\epsilon} \rfloor} \text{tr}(\rho) \\
 &\leq (M_2 + (n - 1)(n^{M_2-1} - 1)) \epsilon^{\lfloor \frac{n-1}{N_\epsilon} \rfloor} \text{tr}(\rho) \\
 &\leq (M_2 + (n - 1)(n^{M_2-1} - 1)_+) \epsilon^{\lfloor \frac{n-1}{N_\epsilon} \rfloor} \text{tr}(\rho),
 \end{aligned} \tag{D.20}$$

where $(x)_+ = \max\{0, x\}$. This inequality also holds for $M_2 = 0$, because if $M_2 = 0$, then

$$\Gamma_\theta^{((n+1)^{M_2} M_1 - 1)} \setminus \Gamma_\theta^{(n^{M_2} M_1 - 1)} = \Gamma_\theta^{((n+1)^0 M_1 - 1)} \setminus \Gamma_\theta^{(n^0 M_1 - 1)} = \Gamma_\theta^{(M_1 - 1)} \setminus \Gamma_\theta^{(M_1 - 1)}$$

is an empty set and $P(\theta)$ does not contain **while** statement, then $\Pi_{S_{P(\theta)}} = \Gamma_\theta^{(M_1)}$.

With Inequality (D.18) and Inequality (D.20), we have that for $n \geq 1$,

$$\begin{aligned}
 &\sum_{\pi \in \Gamma_\theta^{((n+1)^{M_2} M_1 - 1)}} \left| \text{tr} \left(O_d^2 \otimes O \sum_{\eta \in A_\pi} \mathcal{E}_\eta(\rho) \right) \right| \\
 &\leq \sum_{\pi \in \Gamma_\theta^{((n+1)^{M_2} M_1 - 1)}} \sum_{j=1}^{k_\pi} \frac{2M^2}{\mu(j)} \text{tr}(\mathcal{E}_\pi(\rho)) + \sum_{\pi \in \Gamma_\theta^{((n+1)^{M_2} M_1 - 1)}} \sum_{j=1}^{k_\pi} \frac{2M^2}{\mu(j)} \text{tr}(\mathcal{E}_{\pi_{i_j} \pi_{i_{j+1}} \dots \pi_n}(\sigma_{\pi_{i_j}})) \\
 &\equiv \text{Term-A} + \text{Term-B}.
 \end{aligned}$$

For Term-A:

$$\begin{aligned}
 &\sum_{\pi \in \Gamma_\theta^{((n+1)^{M_2} M_1 - 1)}} \sum_{j=1}^{k_\pi} \frac{2M^2}{\mu(j)} \text{tr}(\mathcal{E}_\pi(\rho)) \\
 &\leq \left(\sum_{\pi \in \Gamma_\theta^{(1^{M_2} M_1)}} \sum_{j=1}^{k_\pi} \frac{2M^2}{\mu(j)} \text{tr}(\mathcal{E}_\pi(\rho)) \right) + \left(\sum_{k=1}^n \sum_{\pi \in \Gamma_\theta^{((k+1)^{M_2} M_1 - 1)} \setminus \Gamma_\theta^{((k)^{M_2} M_1 - 1)}} \sum_{j=1}^{k_\pi} \frac{2M^2}{\mu(j)} \text{tr}(\mathcal{E}_\pi(\rho)) \right)
 \end{aligned}$$

¹⁰In the proof of Lemma D.11, we define the set $A_j^{(n,m)}$ for $0 \leq m \leq (n+1)^{M_2} M_1 - 1$. However, we can prove that for any $m > l(k_j)$ and any $\pi \in A_j^{n,m}$, there exists j' such that $k_{j'} < k_j$ and $\pi \in A_{j'}^{n,m'}$ and $m' \leq l(k_{j'})$ by induction on k_j . Thus, we only need consider those sets $A_j^{(n,m)}$ with $0 \leq m \leq l(k_j)$.

$$\begin{aligned}
&\leq \left(\sum_{j=1}^{M_1} \sum_{\pi \in \Gamma_{\theta}^{(M_1)}} \frac{2M^2}{\mu(j)} \text{tr}(\mathcal{E}_{\pi}(\rho)) \right) + \left(\sum_{k=1}^n \sum_{j=1}^{(k+1)^{M_2} M_1 - 1} \sum_{\pi \in \Gamma_{\theta}^{((k+1)^{M_2} M_1 - 1)} \setminus \Gamma_{\theta}^{((k)^{M_2} M_1 - 1)}} \frac{2M^2}{\mu(j)} \text{tr}(\mathcal{E}_{\pi}(\rho)) \right) \\
&\leq \left(\sum_{j=1}^{M_1} \frac{2M^2}{\mu(j)} \text{tr}(\rho) \right) + \left(\sum_{k=1}^n \sum_{j=1}^{(k+1)^{M_2} M_1 - 1} \frac{2M^2}{\mu(j)} (M_2 + (k-1)(k^{M_2-1} - 1)_+) \epsilon^{\lfloor \frac{k-1}{N_{\epsilon}} \rfloor} \text{tr}(\rho) \right) \\
&\quad \text{(By Inequality (D.20))} \\
&\leq 2M^2 \left(S(M_1) + \sum_{k=1}^n \left((M_2 + (k-1)(k^{M_2-1} - 1)_+) S \left((k+1)^{M_2} M_1 - 1 \right) \epsilon^{\lfloor \frac{k-1}{N_{\epsilon}} \rfloor} \right) \right) \text{tr}(\rho),
\end{aligned}$$

where $S(n) = \sum_{j=1}^n \frac{1}{\mu(j)}$.

For Term-B:

$$\begin{aligned}
&\sum_{\pi \in \Gamma_{\theta}^{((n+1)^{M_2} M_1 - 1)}} \sum_{j=1}^{k_{\pi}} \frac{2M^2}{\mu(j)} \text{tr}(\mathcal{E}_{\pi_{i_j} \pi_{i_j+1} \dots \pi_n}(\sigma_{\pi_{i_j}})) \\
&= \left(\sum_{\pi \in \Gamma_{\theta}^{(M_1)}} \sum_{j=1}^{k_{\pi}} \frac{2M^2}{\mu(j)} \text{tr}(\mathcal{E}_{\pi_{i_j} \pi_{i_j+1} \dots \pi_n}(\sigma_{\pi_{i_j}})) \right) \\
&\quad + \left(\sum_{k=1}^n \sum_{\pi \in \Gamma_{\theta}^{((k+1)^{M_2} M_1 - 1)} \setminus \Gamma_{\theta}^{((k)^{M_2} M_1 - 1)}} \sum_{j=1}^{k_{\pi}} \frac{2M^2}{\mu(j)} \text{tr}(\mathcal{E}_{\pi_{i_j} \pi_{i_j+1} \dots \pi_n}(\sigma_{\pi_{i_j}})) \right) \\
&\equiv \text{Term-C} + \text{Term-D}.
\end{aligned}$$

For Term-C, we consider all the $\sigma_{\pi_{i_j}} = \text{tr}_{\bar{q}_{i_j}}(\mathcal{E}_{\pi_1 \dots \pi_{i_j-1}}(\rho)) \otimes \sigma_{i_j}$ that have same $\mathcal{E}_{\pi_1 \dots \pi_{i_j-1}}(\rho)$, let

$$E_j = \left\{ \pi_1 \dots \pi_{i_j-1} : \exists \eta \text{ s.t. } \pi = \pi_1 \dots \pi_{i_j-1} \eta \in \Gamma_{\theta}^{(M_1)} \right\}$$

for every $1 \leq j \leq M_1$ and

$$F_{\pi_1 \dots \pi_{i_j-1}} = \left\{ \pi \in \Gamma_{\theta}^{(M_1)} : \exists \eta \text{ s.t. } \pi = \pi_1 \dots \pi_{i_j-1} \eta \right\}$$

then,

$$\begin{aligned}
&\sum_{\pi \in \Gamma_{\theta}^{(M_1)}} \sum_{j=1}^{k_{\pi}} \frac{2M^2}{\mu(j)} \text{tr}(\mathcal{E}_{\pi_{i_j} \pi_{i_j+1} \dots \pi_n}(\sigma_{\pi_{i_j}})) \\
&= \sum_{j=1}^{M_1} \sum_{\pi_1 \dots \pi_{i_j-1} \in E_j} \sum_{\pi \in F_{\pi_1 \dots \pi_{i_j-1}}} \frac{2M^2}{\mu(j)} \text{tr}(\mathcal{E}_{\pi_{i_j} \pi_{i_j+1} \dots \pi_n}(\sigma_{\pi_{i_j}})) \\
&\leq \sum_{j=1}^{M_1} \sum_{\pi_1 \dots \pi_{i_j-1} \in E_j} \frac{2M^2}{\mu(j)} \text{tr}(\sigma_{\pi_{i_j}}) \quad \text{(apply Lemma D.5 to } F_{\pi_1 \dots \pi_{i_j-1}}) \\
&= \sum_{j=1}^{M_1} \sum_{\pi_1 \dots \pi_{i_j-1} \in E_j} \frac{2M^2}{\mu(j)} \text{tr}(\text{tr}_{\bar{q}_{i_j}}(\mathcal{E}_{\pi_1 \dots \pi_{i_j-1}}(\rho)) \otimes \sigma_{i_j})
\end{aligned}$$

$$\begin{aligned}
&= \sum_{j=1}^{M_1} \sum_{\pi_1 \dots \pi_{i_j-1} \in E_j} \frac{2M^2}{\mu(j)} \text{tr}(\mathcal{E}_{\pi_1 \dots \pi_{i_j-1}}(\rho)) \\
&\leq \sum_{j=1}^{M_1} \frac{2M^2}{\mu(j)} \text{tr}(\rho) \quad (\text{apply Lemma D.5 to } E_j) \\
&= 2M^2 S(M_1) \text{tr}(\rho).
\end{aligned}$$

For Term- D , we have already known that any

$$\pi \in \Gamma_{\theta}^{((k+1)^{M_2} M_1 - 1)} \setminus \Gamma_{\theta}^{((k)^{M_2} M_1 - 1)}$$

must continuously runs into a loop body $P_k(\theta)$, $1 \leq k \leq M_2$ at least n times. For any π above, $\mathcal{E}_{\pi_{i_j} \pi_{i_j+1} \dots \pi_n}(\sigma_{\pi_{i_j}})$ is

$$\mathcal{E}_{\pi_{i_j} \pi_{i_j+1} \dots \pi_n}(\text{tr}_{\tilde{q}_{i_j}}(\mathcal{E}_{\pi_1 \dots \pi_{i_j-1}}(\rho)) \otimes \sigma_{i_j}).$$

For any $1 \leq j \leq ((k+1)^{M_2} M_1 - 1)$, the location π_{i_j} will only split successive runs of the loop body at most once, thus, we can get a scale $\epsilon^{\frac{n-1}{N_{\epsilon}} - 1}$ to replace $\epsilon^{\lfloor \frac{n-1}{N_{\epsilon}} \rfloor}$ in Inequality (D.20) by Lemma 5.3 and Lemma D.13:

$$\begin{aligned}
&\sum_{\pi \in \Gamma_{\theta}^{((k+1)^{M_2} M_1 - 1)} \setminus \Gamma_{\theta}^{(k^{M_2} M_1 - 1)}} \text{tr}(\rho_{\pi, j}) \\
&\leq (M_2 + (k-1)(k^{M_2-1} - 1)_+) \epsilon^{\frac{n-1}{N_{\epsilon}} - 1} \text{tr}(\rho),
\end{aligned} \tag{D.21}$$

where $\rho_{\pi, j} = \mathcal{E}_{\pi_{i_j} \pi_{i_j+1} \dots \pi_n}(\sigma_{\pi_{i_j}})$ if π has j th occurrence of θ , or equal to $\mathcal{E}_{\pi}$ if π has no j th occurrence of θ .

Thus,

$$\begin{aligned}
&\sum_{\pi \in \Gamma_{\theta}^{((k+1)^{M_2} M_1 - 1)} \setminus \Gamma_{\theta}^{((k)^{M_2} M_1 - 1)}} \sum_{j=1}^{k_{\pi}} \frac{2M^2}{\mu(j)} \text{tr}(\mathcal{E}_{\pi_{i_j} \pi_{i_j+1} \dots \pi_n}(\sigma_{\pi_{i_j}})) \\
&= \sum_{\pi \in \Gamma_{\theta}^{((k+1)^{M_2} M_1 - 1)} \setminus \Gamma_{\theta}^{((k)^{M_2} M_1 - 1)}} \sum_{j=1}^{k_{\pi}} \frac{2M^2}{\mu(j)} \text{tr}(\rho_{\pi, j}) \\
&\leq \sum_{\pi \in \Gamma_{\theta}^{((k+1)^{M_2} M_1 - 1)} \setminus \Gamma_{\theta}^{((k)^{M_2} M_1 - 1)}} \sum_{j=1}^{((k+1)^{M_2} M_1 - 1)} \frac{2M^2}{\mu(j)} \text{tr}(\rho_{\pi, j}) \\
&= \sum_{j=1}^{((k+1)^{M_2} M_1 - 1)} \sum_{\pi \in \Gamma_{\theta}^{((k+1)^{M_2} M_1 - 1)} \setminus \Gamma_{\theta}^{((k)^{M_2} M_1 - 1)}} \frac{2M^2}{\mu(j)} \text{tr}(\rho_{\pi, j}) \\
&\leq \sum_{j=1}^{((k+1)^{M_2} M_1 - 1)} \frac{2M^2}{\mu(j)} (M_2 + (k-1)(k^{M_2-1} - 1)_+) \epsilon^{\lfloor \frac{k-1}{N_{\epsilon}} \rfloor - 1} \text{tr}(\rho) \quad (\text{by Inequality (D.21)}) \\
&= 2M^2 (M_2 + (k-1)(k^{M_2-1} - 1)_+) S((k+1)^{M_2} M_1 - 1) \epsilon^{\lfloor \frac{k-1}{N_{\epsilon}} \rfloor - 1} \text{tr}(\rho).
\end{aligned}$$

Then, Term- D can be bounded as follows:

$$\begin{aligned} & \left(\sum_{k=1}^n \sum_{\pi \in \Gamma_{\theta}^{((k+1)^{M_2} M_1 - 1)} \setminus \Gamma_{\theta}^{((k)^{M_2} M_1 - 1)}} \sum_{j=1}^{k_{\pi}} \frac{2M^2}{\mu(j)} \text{tr}(\mathcal{E}_{\pi_{i_j} \pi_{i_{j+1}} \cdots \pi_n}(\sigma_{\pi_{i_j}})) \right) \\ & \leq 2M^2 \sum_{k=1}^n \left((M_2 + (k-1)(k^{M_2-1} - 1)_+) S((k+1)^{M_2} M_1 - 1) \epsilon^{\lfloor \frac{k-1}{N\epsilon} \rfloor - 1} \right) \text{tr}(\rho). \end{aligned}$$

Therefore, Term- B is bounded as follows:

$$\begin{aligned} & \sum_{\pi \in \Gamma_{\theta}^{((n+1)^{M_2} M_1 - 1)}} \sum_{j=1}^{k_{\pi}} \frac{2M^2}{\mu(j)} \text{tr}(\mathcal{E}_{\pi_{i_j} \pi_{i_{j+1}} \cdots \pi_n}(\sigma_{\pi_{i_j}})) \\ & \leq 2M^2 \left(S(M_1) + \sum_{k=1}^n \left((M_2 + (k-1)(k^{M_2-1} - 1)_+) S((k+1)^{M_2} M_1 - 1) \epsilon^{\lfloor \frac{k-1}{N\epsilon} \rfloor - 1} \right) \right) \text{tr}(\rho). \end{aligned}$$

Finally, we have the following inequality:

$$\begin{aligned} & \sum_{\pi \in \Gamma_{\theta}^{((n+1)^{M_2} M_1 - 1)}} \left| \text{tr} \left(O_d^2 \otimes O \sum_{\eta \in A_{\pi}} \mathcal{E}_{\eta}(\rho) \right) \right| \\ & \leq M^2 \left(4S(M_1) + \sum_{k=1}^n \left((M_2 + (k-1)(k^{M_2-1} - 1)_+) \right. \right. \\ & \quad \left. \left. \cdot S((k+1)^{M_2} M_1 - 1) (2\epsilon^{\lfloor \frac{k-1}{N\epsilon} \rfloor} + 2\epsilon^{\lfloor \frac{k-1}{N\epsilon} \rfloor - 1}) \right) \right) \text{tr}(\rho) \end{aligned} \quad (\text{D.22})$$

that holds for any $n \geq 1$ and $M_1, M_2 \geq 0$.

By the definition of $\Gamma_{\theta}^{(n)}$, we have

$$\Pi_{S_{P(\theta)}} = \bigcup_{n=1}^{\infty} \Gamma_{\theta}^{(n)}.$$

When $M_2 \geq 1$, we also have

$$\Pi_{S_{P(\theta)}} = \bigcup_{n=1}^{\infty} \Gamma_{\theta}^{(n^{M_2} M_1 - 1)}.$$

With $\forall n \geq 1, \Gamma_{\theta}^{(n)} \subseteq \Gamma_{\theta}^{(n+1)}$ and Inequality (D.22), we have

$$\begin{aligned} & \left| \sum_{\pi \in \Pi_{S_{P(\theta)}}} \text{tr} \left(O_d^2 \otimes O^2 \sum_{\eta \in A_{\pi}} \mathcal{E}_{\eta}(\rho) \right) \right| = \left| \sum_{\pi \in \bigcup_{n=1}^{\infty} \Gamma_{\theta}^{(n^{M_2} M_1 - 1)}} \text{tr} \left(O_d^2 \otimes O^2 \sum_{\eta \in A_{\pi}} \mathcal{E}_{\eta}(\rho) \right) \right| \\ & \leq \sum_{\pi \in \bigcup_{n=1}^{\infty} \Gamma_{\theta}^{(n^{M_2} M_1 - 1)}} \left| \text{tr} \left(O_d^2 \otimes O^2 \sum_{\eta \in A_{\pi}} \mathcal{E}_{\eta}(\rho) \right) \right| \\ & \leq \lim_{n \rightarrow \infty} \sum_{\pi \in \Gamma_{\theta}^{(n^{M_2} M_1 - 1)}} \left| \text{tr} \left(O_d^2 \otimes O^2 \sum_{\eta \in A_{\pi}} \mathcal{E}_{\eta}(\rho) \right) \right| \end{aligned}$$

$$\begin{aligned}
&= \lim_{n \rightarrow \infty} M^2 \left(4S(M_1) + \sum_{k=1}^n \left((M_2 + (k-1)(k^{M_2-1} - 1)_+) \right. \right. \\
&\quad \left. \left. \cdot S \left((k+1)^{M_2} M_1 - 1 \right) \left(2\epsilon^{\lfloor \frac{k-1}{N\epsilon} \rfloor} + 2\epsilon^{\lfloor \frac{k-1}{N\epsilon} \rfloor - 1} \right) \right) \right) \text{tr}(\rho) \\
&= M^2 \left(4S(M_1) + \sum_{k=1}^{\infty} \left((M_2 + (k-1)(k^{M_2-1} - 1)_+) \right. \right. \\
&\quad \left. \left. \cdot S \left((k+1)^{M_2} M_1 - 1 \right) \left(2\epsilon^{\lfloor \frac{k-1}{N\epsilon} \rfloor} + 2\epsilon^{\lfloor \frac{k-1}{N\epsilon} \rfloor - 1} \right) \right) \right) \\
&\quad \text{(by } \text{tr}(\rho) \leq 1 \text{)}
\end{aligned}$$

When $M_2 = 0$, we already know $\Pi_{S_{P(\theta)}} = \Gamma_{\theta}^{(M_1)}$, then the above inequality also holds.

An equation similar to Equation (D.10) is

$$\langle O_d^2 \otimes O^2 \rangle = \text{tr} \left(O^2 \otimes O_c^2 \left[\left[\frac{\partial}{\partial \alpha} \theta(P(\theta)) \right] (\rho) \right) = \sum_{\pi \in \Pi_{S_{P(\theta)}}} \text{tr} \left(O_d^2 \otimes O^2 \sum_{\eta \in A_{\pi}} \mathcal{E}_{\eta}(\rho) \right).
\right.$$

Thus, we finally obtain the following inequality:

$$\begin{aligned}
&\langle O_d^2 \otimes O^2 \rangle \\
&\leq M^2 \left(4S(M_1) + \sum_{k=1}^{\infty} \left((M_2 + (k-1)(k^{M_2-1} - 1)_+) S \left((k+1)^{M_2} M_1 - 1 \right) \left(2\epsilon^{\lfloor \frac{k-1}{N\epsilon} \rfloor} + 2\epsilon^{\lfloor \frac{k-1}{N\epsilon} \rfloor - 1} \right) \right) \right).
\end{aligned}$$

□

ACKNOWLEDGMENTS

We thank Shenghua Feng for insightful discussions and anonymous reviewers for constructive suggestions that improve the presentation of the article.

CODE AVAILABILITY

The code for implementation (Section 6) and experiments (Section 7) are available at <https://github.com/njuwfang/DifferentiableQPL>

REFERENCES

- [1] Martín Abadi and Gordon D. Plotkin. 2020. A simple differentiable programming language. *Proc. ACM Program. Lang.* 4, POPL (2020), 38:1–38:28. DOI: <https://doi.org/10.1145/3371106>
- [2] Andris Ambainis, Eric Bach, Ashwin Nayak, Ashvin Vishwanath, and John Watrous. 2001. One-dimensional quantum walks. In *Proceedings of the 33rd Annual ACM Symposium on Theory of Computing (STOC'01)*. Association for Computing Machinery, New York, NY, 37–49. DOI: <https://doi.org/10.1145/380752.380757>
- [3] Andris Ambainis, Julia Kempe, and Alexander Rivosh. 2005. Coins make quantum walks faster. In *Proceedings of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'05)*. Society for Industrial and Applied Mathematics, 1099–1108.
- [4] Pablo Andrés-Martínez and Chris Heunen. 2022. Weakly measured while loops: Peeking at quantum states. *Quant. Sci. Technol.* 7, 2 (Feb. 2022), 025007. DOI: <https://doi.org/10.1088/2058-9565/ac47f1>
- [5] Leonardo Banchi and Gavin E. Crooks. 2021. Measuring analytic gradients of general quantum evolution with the stochastic parameter shift rule. *Quantum* 5 (Jan. 2021), 386. DOI: <https://doi.org/10.22331/q-2021-01-25-386>
- [6] Paul Barry. 2011. Eulerian polynomials as moments, via exponential Riordan arrays. *J. Integer Seq.* 14 (05 2011), Article 11.9.5.

- [7] Atilim Gunes Baydin, Barak A. Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind. 2017. Automatic differentiation in machine learning: A survey. *J. Mach. Learn. Res.* 18 (2017), 153:1–153:43. Retrieved from <http://jmlr.org/papers/v18/17-468.html>
- [8] Thomas Beck and Herbert Fischer. 1994. The if-problem in automatic differentiation. *J. Comput. Appl. Math.* 50, 1-3 (1994), 119–131.
- [9] Kerstin Beer, Dmytro Bondarenko, Terry Farrelly, Tobias J. Osborne, Robert Salzmänn, Daniel Scheiermann, and Ramona Wolf. 2020. Training deep quantum neural networks. *Nat. Commun.* 11, 1 (12 2020), 1–6. DOI : <https://doi.org/10.1038/s41467-020-14454-2>
- [10] Marcello Benedetti, Erika Lloyd, Stefan Sack, and Mattia Fiorentini. 2019. Parameterized quantum circuits as machine learning models. *Quant. Sci. Technol.* 4, 4 (11 2019), 19601. DOI : <https://doi.org/10.1088/2058-9565/ab4eb5>
- [11] Paul Benioff. 2000. Space searches with a quantum robot. *arXiv preprint quant-ph/0003006* (2000).
- [12] Ville Bergholm, Josh Izaac, Maria Schuld, Christian Gogolin, M. Sohaib Alam, Shahnawaz Ahmed, Juan Miguel Arzola, Carsten Blank, Alain Delgado, Soran Jahangiri, Keri McKiernan, Johannes Jakob Meyer, Zeyue Niu, Antal Száva, and Nathan Killoran. 2020. PennyLane: Automatic Differentiation of Hybrid Quantum-classical Computations. *arXiv:1811.04968* [quant-ph]
- [13] Benjamin Bichsel, Maximilian Baader, Timon Gehr, and Martin Vechev. 2020. Silq: A high-level quantum language with safe uncomputation and intuitive semantics. In *Proceedings of the 41st ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI'20)*. Association for Computing Machinery, New York, NY, 286–300. DOI : <https://doi.org/10.1145/3385412.3386007>
- [14] Alex Bocharov, Martin Roetteler, and Krysta M. Svore. 2015. Efficient synthesis of universal repeat-until-success quantum circuits. *Phys. Rev. Lett.* 114, 8 (Feb. 2015), 080502. DOI : <https://doi.org/10.1103/PhysRevLett.114.080502>
- [15] Gilles Brassard, Peter Hoyer, Michele Mosca, and Alain Tapp. 2002. Quantum amplitude amplification and estimation. *Contemp. Math.* 305 (2002), 53–74.
- [16] Michael Broughton, Guillaume Verdon, Trevor McCourt, Antonio J. Martinez, Jae Hyeon Yoo, Sergei V. Isakov, Philip Massey, Murphy Yuezhen Niu, Ramin Halavati, Evan Peters, Martin Leib, Andrea Skolik, Michael Streif, David Von Dollen, Jarrod R. McClean, Sergio Boixo, Dave Bacon, Alan K. Ho, Hartmut Neven, and Masoud Mohseni. 2020. TensorFlow Quantum: A Software Framework for Quantum Machine Learning. *arXiv:2003.02989* [quant-ph]
- [17] Andrew M. Childs. 2009. On the relationship between continuous- and discrete-time quantum walk. *Commun. Math. Phys.* 294, 2 (Oct. 2009), 581–603. DOI : <https://doi.org/10.1007/s00220-009-0930-1>
- [18] Andrew M. Childs and Nathan Wiebe. 2012. Hamiltonian simulation using linear combinations of unitary operations. *Quant. Inf. Comput.* 12, 11-12 (2012), 901–924. DOI : <https://doi.org/10.26421/QIC12.11-12-1>
- [19] Iris Cong, Soonwon Choi, and Mikhail D. Lukin. 2019. Quantum convolutional neural networks. *Nat. Phys.* 15, 12 (12 2019), 1273–1278. DOI : <https://doi.org/10.1038/s41567-019-0648-8>
- [20] George Corliss, Christèle Faure, Andreas Griewank, Lauren Hascoët, and Uwe Naumann (Eds.). 2002. *Automatic Differentiation of Algorithms: From Simulation to Optimization*. Springer-Verlag New York, Inc., New York, NY.
- [21] Olivia Di Matteo, Josh Izaac, Tom Bromley, Anthony Hayes, Christina Lee, Maria Schuld, Antal Száva, Chase Roberts, and Nathan Killoran. 2022. Quantum computing with differentiable quantum transforms. *arXiv preprint arXiv:2202.13414* (2022).
- [22] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. 2014. A Quantum Approximate Optimization Algorithm. *arXiv:1411.4028* [quant-ph]
- [23] Edward Farhi and Aram W. Harrow. 2016. Quantum Supremacy through the Quantum Approximate Optimization Algorithm. *arXiv:1602.07674* [quant-ph]
- [24] Edward Farhi and Hartmut Neven. 2018. Classification with Quantum Neural Networks on Near Term Processors. *arXiv:1802.06002* [quant-ph]
- [25] Alexander L. Gaunt, Marc Brockschmidt, Nate Kushman, and Daniel Tarlow. 2017. Differentiable programs with neural libraries. In *Proceedings of the 34th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 70)*, Doina Precup and Yee Whye Teh (Eds.). PMLR, 1213–1222. Retrieved from <http://proceedings.mlr.press/v70/gaunt17a.html>
- [26] Alex Graves, Greg Wayne, Malcolm Reynolds, Tim Harley, Ivo Danihelka, Agnieszka Grabska-Barwinska, Sergio Gomez Colmenarejo, Edward Grefenstette, Tiago Ramalho, John P. Agapiou, Adrià Puigdomènech Badia, Karl Moritz Hermann, Yori Zwols, Georg Ostrovski, Adam Cain, Helen King, Christopher Summerfield, Phil Blunsom, Koray Kavukcuoglu, and Demis Hassabis. 2016. Hybrid computing using a neural network with dynamic external memory. *Nature* 538, 7626 (2016), 471–476. DOI : <https://doi.org/10.1038/nature20101>
- [27] Edward Grefenstette, Karl Moritz Hermann, Mustafa Suleyman, and Phil Blunsom. 2015. Learning to transduce with unbounded memory. In *Proceedings of the 28th International Conference on Neural Information Processing Systems (NIPS'15)*. MIT Press, Cambridge, MA, 1828–1836. Retrieved from <http://dl.acm.org/citation.cfm?id=2969442.2969444>

- [28] Andreas Griewank. 2000. *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation*. Society for Industrial and Applied Mathematics, Philadelphia, PA.
- [29] Lov K. Grover. 1996. A fast quantum mechanical algorithm for database search. In *Proceedings of the 28th Annual ACM Symposium on Theory of Computing (STOC'96)*. Association for Computing Machinery, New York, NY, 212–219. DOI: <https://doi.org/10.1145/237814.237866>
- [30] Ji Guan, Yuan Feng, and Mingsheng Ying. 2018. Decomposition of quantum Markov chains and its applications. *J. Comput. Syst. Sci.* 95 (2018), 55–68. DOI: <https://doi.org/10.1016/j.jcss.2018.01.005>
- [31] Gian Giacomo Guerreschi and Mikhail Smelyanskiy. 2017. Practical Optimization for Hybrid Quantum-classical Algorithms. arXiv:1701.01450 [quant-ph]
- [32] Stuart Hadfield, Zhihui Wang, Bryan O’Gorman, Eleanor Gilbert Rieffel, Davide Venturelli, and Rupak Biswas. 2019. From the quantum approximate optimization algorithm to a quantum alternating operator ansatz. *Algorithms* 12, 2 (2019), 34. DOI: <https://doi.org/10.3390/a12020034>
- [33] Godefroy Harold Hardy. 1916. Weierstrass’s non-differentiable function. *Trans. Am. Math. Soc* 17, 3 (1916), 301–325.
- [34] Mathieu Huot, Sam Staton, and Matthijs Vákár. 2020. Correctness of automatic differentiation via diffeologies and categorical gluing. In *Proceedings of the 23rd International Conference on Foundations of Software Science and Computation Structures (FOSSACS’20), Held as Part of the European Joint Conferences on Theory and Practice of Software (ETAPS’20) (Lecture Notes in Computer Science, Vol. 12077)*, Jean Goubault-Larrecq and Barbara König (Eds.). Springer, Cham, 319–338. DOI: https://doi.org/10.1007/978-3-030-45231-5_17
- [35] Artur F. Izmaylov, Robert A. Lang, and Tzu-Ching Yen. 2021. Analytic gradients in variational quantum algorithms: Algebraic extensions of the parameter-shift rule to general unitary transformations. *Phys. Rev. A* 104, 6 (2021), 062443.
- [36] Abhinav Kandala, Antonio Mezzacapo, Kristan Temme, Maika Takita, Markus Brink, Jerry M. Chow, and Jay M. Gambetta. 2017. Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets. *Nature* 549, 7671 (9 2017), 242–246. DOI: <https://doi.org/10.1038/nature23879>
- [37] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR’15)*, Yoshua Bengio and Yann LeCun (Eds.). Retrieved from <http://arxiv.org/abs/1412.6980>
- [38] Jakob S. Kottmann, Sumner Alperin-Lea, Teresa Tamayo-Mendoza, Alba Cervera-Lierta, Cyrille Lavigne, Tzu-Ching Yen, Vladyslav Verteletskyi, Philipp Schleich, Abhinav Anand, Matthias Degroote, Skylar Chaney, Maha Kesibi, Naomi Grace Curnow, Brandon Solo, Georgios Tsilimigkounakis, Claudia Zendejas-Morales, Artur F. Izmaylov, and Alán Aspuru-Guzik. 2021. Tequila: A platform for rapid development of quantum algorithms. *Quant. Sci. Technol.* 6, 2 (2021), 024009. <https://dx.doi.org/10.1088/2058-9565/abe567>
- [39] Oleksandr Kyriienko and Vincent E. Elfving. 2021. Generalized quantum circuit differentiation rules. *Phys. Rev. A* 104, 5 (2021), 052417.
- [40] Wonyeol Lee, Hangeol Yu, Xavier Rival, and Hongseok Yang. 2019. Towards verified stochastic variational inference for probabilistic programs. *Proc. ACM Program. Lang.* 4, POPL, Article 16 (Dec. 2019), 33 pages. DOI: <https://doi.org/10.1145/3371084>
- [41] Alexander K. Lew, Mathieu Huot, Sam Staton, and Vikash K. Mansinghka. 2023. ADEV: Sound automatic differentiation of expected values of probabilistic programs. *Proc. ACM Program. Lang.* 7, POPL (2023), 121–153.
- [42] Jun Li, Xiaodong Yang, Xinhua Peng, and Chang-Pu Sun. 2017. Hybrid quantum-classical approach to quantum optimal control. *Phys. Rev. Lett.* 118, 15 (Apr. 2017), 150503. DOI: <https://doi.org/10.1103/PhysRevLett.118.150503>
- [43] Yuan Liang Lim, Almut Beige, and Leong Chuan Kwek. 2005. Repeat-until-success linear optics distributed quantum computing. *Phys. Rev. Lett.* 95, 3 (Jul. 2005), 030505. DOI: <https://doi.org/10.1103/PhysRevLett.95.030505>
- [44] Seth Lloyd, Masoud Mohseni, and Patrick Rebentrost. 2014. Quantum principal component analysis. *Nat. Phys.* 10, 9 (Jul. 2014), 631–633. DOI: <https://doi.org/10.1038/nphys3029>
- [45] Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *Proceedings of the 7th International Conference on Learning Representations (ICLR’19)*. OpenReview.net. Retrieved from <https://openreview.net/forum?id=Bkg6RiCqY7>
- [46] Xiu-Zhe Luo, Jin-Guo Liu, Pan Zhang, and Lei Wang. 2020. Yao. jl: Extensible, efficient framework for quantum algorithm design. *Quantum* 4 (2020), 341.
- [47] Carol Mak, C.-H. Luke Ong, Hugo Paquet, and Dominik Wagner. 2021. Densities of almost surely terminating probabilistic programs are differentiable almost everywhere. In *Proceedings of the 30th European Symposium on Programming (ESOP’21), Held as Part of the European Joint Conferences on Theory and Practice of Software (ETAPS’21)*. 432–461. Retrieved from <https://europepmc.org/articles/PMC7984578>
- [48] Damiano Mazza and Michele Pagani. 2021. Automatic differentiation in PCF. *Proc. ACM Program. Lang.* 5, POPL, Article 28 (Jan. 2021), 27 pages. DOI: <https://doi.org/10.1145/3434309>
- [49] K. Mitarai, M. Negoro, M. Kitagawa, and K. Fujii. 2018. Quantum circuit learning. *Phys. Rev. A* 98, 3 (Sep. 2018), 032309. DOI: <https://doi.org/10.1103/PhysRevA.98.032309>

- [50] Ari Mizel. 2009. Critically damped quantum search. *Phys. Rev. Lett.* 102, 15 (Apr. 2009), 150501. DOI : <https://doi.org/10.1103/PhysRevLett.102.150501>
- [51] Quoc Chuong Nguyen, Le Bin Ho, Lan Nguyen Tran, and Hung Q. Nguyen. 2022. Qsun: An open-source platform towards practical quantum machine learning applications. *Mach. Learn.: Sci. Technol.* 3, 1 (2022), 015034. <https://dx.doi.org/10.1088/2632-2153/ac5997>
- [52] Michael A. Nielsen and Isaac L. Chuang. 2011. *Quantum Computation and Quantum Information: 10th Anniversary Edition* (10th ed.). Cambridge University Press.
- [53] Adam Paetznick and Krysta M. Svore. 2014. Repeat-until-success: Non-deterministic decomposition of single-qubit unitaries. *Quant. Inf. Comput.* 14, 15–16 (Nov. 2014), 1277–1301.
- [54] Alberto Peruzzo, Jarrod McClean, Peter Shadbolt, Man Hong Yung, Xiao Qi Zhou, Peter J. Love, Alán Aspuru-Guzik, and Jeremy L. O’Brien. 2014. A variational eigenvalue solver on a photonic quantum processor. *Nat. Commun.* 5, 1 (7 2014), 1–7. DOI : <https://doi.org/10.1038/ncomms5213>
- [55] Gordon Plotkin. 2018. Some principles of differential programming languages. In *Proceedings of the Symposium on Principles of Programming Languages*.
- [56] John Preskill. 2018. Quantum computing in the NISQ era and beyond. *Quantum* 2 (8 2018), 79. DOI : <https://doi.org/10.22331/q-2018-08-06-79>
- [57] Sebastian Ruder. 2017. An Overview of Gradient Descent Optimization Algorithms. arXiv:1609.04747 [cs.LG]
- [58] Walter Rudin. 1976. *Principles of Mathematical Analysis*. Vol. 3. McGraw-Hill.
- [59] Maria Schuld, Ville Bergholm, Christian Gogolin, Josh Izaac, and Nathan Killoran. 2019. Evaluating analytic gradients on quantum hardware. *Phys. Rev. A* 99, 3 (Mar. 2019), 032331. DOI : <https://doi.org/10.1103/PhysRevA.99.032331>
- [60] Benjamin Sherman, Jesse Michel, and Michael Carbin. 2021. λ_s : Computable semantics for differentiable programming with higher-order functions and datatypes. *Proc. ACM Program. Lang.* 5, POPL, Article 3 (Jan. 2021), 31 pages. DOI : <https://doi.org/10.1145/3434284>
- [61] Rajan Srinivasan. 2002. *Importance Sampling: Applications in Communications and Detection*. Springer Science & Business Media.
- [62] Krysta Svore, Alan Geller, Matthias Troyer, John Azariah, Christopher Granade, Bettina Heim, Vadym Kliuchnikov, Mariia Mykhailova, Andres Paz, and Martin Roetteler. 2018. Q#: Enabling scalable quantum computing and development with a high-level DSL. In *Proceedings of the Real World Domain Specific Languages Workshop (RWDSL’18)*. Association for Computing Machinery, New York, NY. DOI : <https://doi.org/10.1145/3183895.3183901>
- [63] Ryan Sweke, Frederik Wilde, Johannes Meyer, Maria Schuld, Paul K. Faehrmann, Barthélémy Meynard-Piganeau, and Jens Eisert. 2020. Stochastic gradient descent for hybrid quantum-classical optimization. *Quantum* 4 (Aug. 2020), 314. DOI : <https://doi.org/10.22331/q-2020-08-31-314>
- [64] Mario Szegedy. 2004. Quantum speed-up of Markov chain based algorithms. In *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science (FOCS’04)*. IEEE Computer Society, 32–41. DOI : <https://doi.org/10.1109/FOCS.2004.53>
- [65] Javier Gil Vidal and Dirk Oliver Theis. 2018. Calculus on Parameterized Quantum Circuits. arXiv:1812.06323 [quant-ph]
- [66] Fei Wang, Daniel Zheng, James Decker, Xilun Wu, Grégory M. Essertel, and Tiark Rompf. 2019. Demystifying differentiable programming: Shift/reset the penultimate backpropagator. *Proc. ACM Program. Lang.* 3, ICFP (7 2019), 1–31. DOI : <https://doi.org/10.1145/3341700>
- [67] Nathan Wiebe and Martin Roetteler. 2016. Quantum arithmetic and numerical analysis using repeat-until-success circuits. *Quant. Inf. Comput.* 16, 1–2 (Jan. 2016), 134–178.
- [68] David Wierichs, Josh Izaac, Cody Wang, and Cedric Yen-Yu Lin. 2022. General parameter-shift rules for quantum gradients. *Quantum* 6 (2022), 677.
- [69] Michael M. Wolf. 2012. Quantum Channels & Operations: Guided Tour. Lecture Notes. Retrieved from <https://www-m5.ma.tum.de/foswiki/pub/M5/Allgemeines/MichaelWolf/QChannelLecture.pdf>
- [70] Thomas G. Wong. 2017. Equivalence of Szegedy’s and coined quantum walks. *Quant. Inf. Process.* 16, 9 (Jul. 2017). DOI : <https://doi.org/10.1007/s11228-017-1667-y>
- [71] W. K. Wootters and W. H. Zurek. 1982. A single quantum cannot be cloned. *Nature* 299, 5886 (1982), 802–803. DOI : <https://doi.org/10.1038/299802a0>
- [72] Hongseok Yang. 2019. Some semantic issues in probabilistic programming languages (invited talk). In *Proceedings of the 4th International Conference on Formal Structures for Computation and Deduction (FSCD’19)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.
- [73] Mingsheng Ying. 2016. *Foundations of Quantum Programming*. Morgan Kaufmann.
- [74] Mingsheng Ying, Shenggang Ying, and Xiaodi Wu. 2017. Invariants of quantum programs: Characterisations and generation. In *Proceedings of the 44th ACM SIGPLAN Symposium on Principles of Programming Languages (POPL’17)*. Association for Computing Machinery, New York, NY, 818–832. DOI : <https://doi.org/10.1145/3009837.3009840>

- [75] Mingsheng Ying, Nengkun Yu, Yuan Feng, and Runyao Duan. 2013. Verification of quantum programs. *Sci. Comput. Program.* 78, 9 (2013), 1679–1700. DOI : <https://doi.org/10.1016/j.scico.2013.03.016>
- [76] Shenggang Ying, Yuan Feng, Nengkun Yu, and Mingsheng Ying. 2013. Reachability probabilities of quantum Markov chains. In *Proceedings of the 24th International Conference on Concurrency Theory (CONCUR'13) (Lecture Notes in Computer Science*, Vol. 8052), Pedro R. D'Argenio and Hernán C. Melgratti (Eds.). Springer, Berlin, 334–348. DOI : https://doi.org/10.1007/978-3-642-40184-8_24
- [77] Shaopeng Zhu, Shih-Han Hung, Shouvanik Chakrabarti, and Xiaodi Wu. 2020. On the principles of differentiable quantum programming languages. In *Proceedings of the 41st ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI'20)*. Association for Computing Machinery, New York, NY, 272–285. DOI : <https://doi.org/10.1145/3385412.3386011>

Received 26 March 2023; revised 5 July 2023; accepted 24 July 2023