# Absence of Barren Plateaus in Quantum Convolutional Neural Networks

Arthur Pesah[1,2] M. Cerezo,[1,3] Samson Wang,[1,4] Tyler Volkoff,[1] Andrew T. Sornborger,[5] and Patrick J. Coles[1]

[1]*Theoretical Division, Los Alamos National Laboratory, Los Alamos, New Mexico 87545, USA*
[2]*Department of Physics and Astronomy, University College London, London WC1E 6BT, United Kingdom*
[3]*Center for Nonlinear Studies, Los Alamos National Laboratory, Los Alamos, New Mexico 87544*
[4]*Imperial College London, London, United Kingdom*
[5]*Information Sciences, Los Alamos National Laboratory, Los Alamos, New Mexico 87544, USA*

Quantum neural networks (QNNs) have generated excitement around the possibility of efficiently analyzing quantum data. But this excitement has been tempered by the existence of exponentially vanishing gradients, known as barren plateau landscapes, for many QNN architectures. Recently, quantum convolutional neural networks (QCNNs) have been proposed, involving a sequence of convolutional and pooling layers that reduce the number of qubits while preserving information about relevant data features. In this work, we rigorously analyze the gradient scaling for the parameters in the QCNN architecture. We find that the variance of the gradient vanishes no faster than polynomially, implying that QCNNs do not exhibit barren plateaus. This result provides an analytical guarantee for the trainability of randomly initialized QCNNs, which highlights QCNNs as being trainable under random initialization unlike many other QNN architectures. To derive our results, we introduce a novel graph-based method to analyze expectation values over Haar-distributed unitaries, which will likely be useful in other contexts. Finally, we perform numerical simulations to verify our analytical results.

Subject Areas: Quantum Information

## I. INTRODUCTION

The field of classical machine learning has been revolutionized by the advent of neural networks (NNs). One of the most prominent forms of classical neural network is the convolutional neural network (CNN) [1,2]. CNNs differ from traditional fully connected neural networks in that they use kernels applied across network layers which reduce the dimension of the data and the number of parameters that need to be trained. The architecture of CNNs was inspired by seminal experiments investigating the structure of the visual cortex [3,4]. Indeed, CNNs have been employed for image-based tasks as their structure allows for local pattern recognition; reviews of state-of-the-art techniques can be found in Refs. [5–7].

Despite the tremendous success of NNs, several difficulties were encountered during their development that hindered the trainability of the NN's parameters. While multilayer perceptrons are more powerful than single-layer ones [8,9], their trainability is more challenging, and the

backpropagation method was developed to address this issue [10]. In addition, training difficulties can also arise in NNs that employ backpropagation and gradient-based methods, as the loss function gradients can be vanishing [11]. For CNNs this phenomenon can be mitigated by choosing rectifier neuron activation functions [12] or by implementing layer-by-layer training.

With the advent of noisy intermediate-scale quantum computers, the field of quantum machine learning has brought new approaches to solving computational problems dealing with quantum data [13–16]. Among the most promising applications are variational quantum algorithms (VQAs) [17–26], and more generally, quantum neural networks (QNNs) [27–34]. In the near term, these architectures employ a noisy hardware to evaluate a cost (or loss) function, while leveraging the power of classical optimizers to train the parameters in a quantum circuit, or a neural network.

Recently, tremendous effort has been put forward to analyze the trainability of the cost functions of VQAs and QNNs, as it has been shown that the optimization landscape can exhibit the so-called *barren plateau* phenomenon where the cost function gradients of a randomly initialized ansatz vanish exponentially with the problem size [35–44]. If the cost exhibits a barren plateau, an exponentially large precision is needed to navigate through the landscape, rendering the architecture unscalable.

Specifically, it has been shown that barren plateaus can arise for different architectures such as VQAs with deep random parametrized quantum circuits [35] or with global cost functions [36]. Additionally, recent studies have shown that more general perceptron-based dissipative QNNs [32] can also show barren plateau landscapes [37,43], proving that this is a general phenomenon in many quantum machine learning applications. While strategies have been proposed to mitigate the effects of barren plateaus and improve trainability [45–54], much work still remains to be done to guarantee the efficient trainability of VQAs and QNNs.

In this work, we analyze the trainability and the existence of barren plateaus in the quantum convolutional neural network (QCNN) architecture introduced by Cong *et al.* in Ref. [34]. A very similar architecture was proposed by Grant *et al.* in Ref. [55] under the name "hierarchical quantum classifier," which our work applies to as well. Motivated by the structure of CNNs, in a QCNN a series of convolutional layers are interleaved with pooling layers which effectively reduce the number of degrees of freedom, while preserving the relevant features of the input state. We remark that due to the specific architecture of the QCNN, its trainability does not follow from previous works. QCNNs have been successfully implemented for error correction, quantum phase detection [34,56], and image recognition [57], and their shallow depth makes them a promising architecture for the near term.

Here we provide a rigorous analysis of the scaling of the QCNN cost function gradient, under the following assumptions: (1) All the two-qubit unitaries in the QCNN form independent (and uncorrelated) 2-designs and (2) the cost function is linear with respect to the input density matrix. The first condition can easily be fulfilled with low-depth circuits, as shown in the Appendix E. The second condition encompasses several types of problems, including classification, as discussed in Sec. II B. Under these assumptions, we show that the variance of the cost function partial derivatives is at most polynomially vanishing with the system size. This implies that the cost function landscape does not exhibit a barren plateau, and hence that the QCNN architecture is trainable under random initialization of parameters. We obtain our results by introducing a novel method to compute the expectation values of operators defined in terms of Haar-distributed unitaries, which we call the graph recursion integration method (GRIM). Intuitively, our results follow from the intrinsically short depth of the QCNN (logarithmic depth) and the local nature of the cost function. We then also provide trainability guarantees for the special case where the QCNN is composed only of pooling layers. Finally, we perform numerical simulations to verify our analytical results.

Our paper is organized as follows: In Sec. II, we start by reviewing the necessary theoretical background on QCNNs, barren plateaus, and Haar-distributed unitaries.

The GRIM is introduced in Sec. III A. Section III B contains our main results in the form of Theorem 1 and concomitant Corollary 1 which show the absence of barren plateaus for randomly initialized QCNNs. Then, in Sec. III C we present Theorem 2 which analyzes the pooling-based QCNN, while Sec. III D contains our numerical results. Finally, our discussion and conclusion are presented in Sec. IV. The proof of our main results are presented in the Appendix.

## II. THEORETICAL FRAMEWORK

### A. The QCNN architecture

As depicted in Fig. 1(a), the QCNN architecture takes as input an $n$-qubit input state $\rho_{\text{in}}$ in a Hilbert space $\mathcal{H}_{\text{in}}$, which is sent through a circuit composed of a sequence of convolutional and pooling layers. The convolutional layer is composed of two rows of parametrized two-qubit gates acting on alternating pairs of neighboring qubits. In each pooling layer, half of the qubits are measured, with the measurement outcomes controlling the unitary applied to neighboring qubits. After $L$ convolutional and pooling layers, the QCNN contains a fully connected layer that applies a unitary to the remaining qubits. Finally, at the end
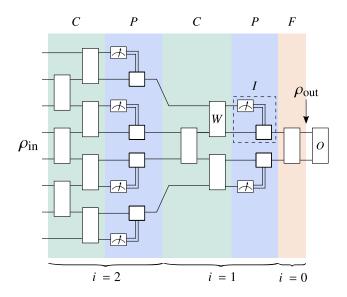


FIG. 1. Schematic representation of a QCNN. The input of the QCNN is an $n$-qubit quantum state $\rho_{\text{in}}$. The state $\rho_{\text{in}}$ is then sent through a sequence of $L$ convolutional ($C$) and pooling ($P$) layers. The convolutional layers are composed of two rows of two-qubit unitaries ($W$) acting on alternating pairs of qubits. The pooling layer is composed of pooling operators (indicated in the dashed box). In each pooling module, a qubit is measured with the measurement outcome controlling a unitary applied to a neighboring qubit ($I$). After the final pooling layer, one applies a fully connected unitary ($F$) to the remaining qubits and obtains an output state $\rho_{\text{out}}$ whose dimension is much smaller than that of $\rho_{\text{in}}$. Finally, one measures the expectation value of some operator $O$ over the state $\rho_{\text{out}}$.

of the QCNN, one measures the expectation value of some Hermitian operator $O$.

From the previous description, we have that the input state to the QCNN is mapped to a reduced state in a Hilbert space $\mathcal{H}_{\text{out}}$ whose dimension is much smaller than that of $\mathcal{H}_{\text{in}}$. The output state can be expressed as

$$\rho_{\text{out}}(\boldsymbol{\theta}) = \text{Tr}_{\overline{\text{out}}}[V(\boldsymbol{\theta})\rho_{\text{in}}V^{\dagger}(\boldsymbol{\theta})]. \quad (1)$$

Here, $V(\boldsymbol{\theta})$ is the unitary that contains the gates in the convolutional and pooling layers plus the fully connected layer, $\boldsymbol{\theta}$ is the vector of the trainable parameters, and $\text{Tr}_{\overline{\text{out}}}$ denotes the partial trace over all qubits except those in $\mathcal{H}_{\text{out}}$. Note that the nonlinearities in a QCNN arise from the pooling operators (measurement and conditioned unitary) in the pooling layers, which effectively reduce the degrees of freedom in each layer.

## B. Cost function

The goal of the QCNN is to employ a training set $\mathcal{S}$ (of size $M = |\mathcal{S}|$) containing input states $\{\rho_{\text{in}}^{\alpha}\}_{\alpha=1}^{M}$ to optimize the parameters in the QCNN and minimize a cost function, which we assume can be expressed as

$$C(\boldsymbol{\theta}) = \sum_{\rho_{\text{in}}^{\alpha} \in \mathcal{S}} c_{\alpha} \text{Tr}[\rho_{\text{out}}^{\alpha}(\boldsymbol{\theta})O], \quad (2)$$

where $c_{\alpha}$ are real coefficients, and where $\rho_{\text{out}}^{\alpha}(\boldsymbol{\theta})$ are obtained from Eq. (1) for each input state $\rho_{\text{in}}$.

For example, in a binary classification problem, the training set is usually of the form $\mathcal{S} = \{\rho_{\text{in}}^{\alpha}, y^{\alpha}\}_{\alpha=1}^{M}$ with $y^{\alpha} = 0$ or $y^{\alpha} = 1$. Here we can divide $\mathcal{S}$ into two subsets $\mathcal{S}_0$ and $\mathcal{S}_1$ of sizes $M_0 = |\mathcal{S}_0|$ and $M_1 = |\mathcal{S}_1|$ (such that $M = M_0 + M_1$), respectively, composed of states associated with outputs 0 and 1. In this case, Eq. (2) can be explicitly expressed as

$$C(\boldsymbol{\theta}) = \sum_{\rho_{\text{in}}^{\alpha} \in \mathcal{S}_0} \frac{\text{Tr}[\rho_{\text{out}}^{\alpha}(\boldsymbol{\theta})O]}{M_0} - \sum_{\rho_{\text{in}}^{\alpha} \in \mathcal{S}_1} \frac{\text{Tr}[\rho_{\text{out}}^{\alpha}(\boldsymbol{\theta})O]}{M_1}, \quad (3)$$

where the operator $O$ is such that for any state $\tau \in \mathcal{H}_{\text{out}}$ we have $\text{Tr}[\tau O] \in [0, 1]$.

Here we remark that it is convenient to rewrite the cost function as

$$C(\boldsymbol{\theta}) = \text{Tr}[V(\boldsymbol{\theta})\sigma V^{\dagger}(\boldsymbol{\theta})\tilde{O}], \quad (4)$$

where $\tilde{O} = (O \otimes \mathbb{1}_{\overline{\text{out}}})$, and $\mathbb{1}_{\overline{\text{out}}}$ is the identity operator over the Hilbert space $\mathcal{H}_{\overline{\text{out}}}$ which is defined such that $\mathcal{H}_{\text{in}} = \mathcal{H}_{\text{out}} \otimes \mathcal{H}_{\overline{\text{out}}}$. In addition, here we define the Hermitian operator

$$\sigma = \sum_{\rho_{\text{in}}^{\alpha} \in \mathcal{S}} c_{\alpha} \rho_{\text{in}}^{\alpha}, \quad (5)$$

which is a quantum state for the special case when $c_{\alpha} \geq 0$ for all $\alpha$, and when $\sum c_{\alpha} = 1$. As we show below, our results are valid independent of $\sigma$ being a quantum state.

## C. Ansatz

In what follows, we consider for simplicity the case where $n = 2^k$ and $L = \log(n) = k$, so that $\dim(\mathcal{H}_{\text{out}}) = 2$. Moreover, we assume that the unitaries in the convolutional and pooling layers are independent. That is, the convolutional and the fully connected layers in $V(\boldsymbol{\theta})$ are composed of two-qubit parametrized unitary blocks acting on neighboring qubits, which we denote as $W_{ij}(\boldsymbol{\theta}_{ij})$. We henceforth use the terminology of sublayers, such that the unitaries in the first sublayer act before those in the second sublayer. On the other hand, the pooling layers are composed of pooling operators $I_{ij}$. Here, $i = 0, ..., L$ is the layer index with $i = 0$ corresponding to the fully connected layer, while $j$ is the index that determines block placement within the layer. We remark that this generalization contains as a special case the usual QCNN structure where the blocks in the same convolutional or pooling layer are identical. Moreover, as we discuss in our results section, correlating the unitaries in the convolutional layers tends to increase the magnitude of the cost function gradient.

Given a two-qubit unitary $W_{ij}(\boldsymbol{\theta}_{ij})$ in a convolutional or fully connected layer, it is common to expand it as a product of two-qubit gates of the form

$$W_{ij} = \prod_{\eta} e^{-i\theta_{\eta}H_{\eta}} W_{\eta}, \quad (6)$$

where $H_{\eta}$ is a Hermitian operator, $e^{-i\theta_{\eta}H_{\eta}}$ is a parametrized gate (such as a rotation), $W_{\eta}$ is an unparametrized gate [such as a controlled NOT (CNOT) gate], and where $\boldsymbol{\theta}_{ij} = (\theta_1, ..., \theta_{\eta}, ...)$. For simplicity of notation, here we omit the parameter dependence in $W_{ij}$. Note that Eq. (6) can be readily used to determine the circuit description of $W_{ij}$. Moreover, if the operator $H_{\eta}$ has eigenvalues $\pm 1$ (or more generally, if it has only two distinct eigenvalues), one can evaluate the cost partial derivative via the parameter-shift rule [58,59].

On the other hand, the modules in the pooling layers can be thought of as a map from a two-qubit Hilbert space $\mathcal{H}_{AB} = \mathcal{H}_A \otimes \mathcal{H}_B$ to a single-qubit Hilbert space $\mathcal{H}_B$. Given a two-qubit state $\tau \in \mathcal{H}_{AB}$, the action of the measurement and the controlled unitary is given by $\tau_2 = \text{Tr}_A[I_{ij}\tau]$, where $\text{Tr}_A$ denotes the partial trace over $\mathcal{H}_A$, and where

$$I_{ij} = (\Pi_0 \otimes U_0^{ij} + \Pi_1 \otimes U_1^{ij}). \quad (7)$$

Here, $\Pi_k = |i\rangle\langle i|$ for $k = 1, 0$, and $U_k^{ij}$ are parametrized single-qubit unitaries. It is straightforward to verify that the operators $I_{ij}$ are unitaries: $I_{ij}I_{ij}^{\dagger} = I_{ij}^{\dagger}I_{ij} = \mathbb{1}$.

### D. Trainability and variance of the cost

Consider a given trainable parameter $\theta_\mu$ in a unitary $W_{ij}$ appearing in the $\ell$th layer of $V(\boldsymbol{\theta})$, which for simplicity we denote as $W$, and let the cost function partial derivative with respect to $\theta_\mu$ be $\partial C(\boldsymbol{\theta})/\partial\theta_\mu = \partial_\mu C$. Then, let us define the two-qubit Hilbert space where $W$ acts on as $\mathcal{H}_w$, such that $\mathcal{H}_{\text{in}} = \mathcal{H}_w \otimes \mathcal{H}_{\bar{w}}$.

As shown in Refs. [35–38], the trainability of the parameters in a randomly initialized QCNN can be analyzed by studying the scaling of the variance

$$\text{Var}[\partial_\mu C] = \langle(\partial_\mu C)^2\rangle - \langle\partial_\mu C\rangle^2, \qquad (8)$$

where the expectation value $\langle\cdots\rangle$ is taken over the parameters in $V(\boldsymbol{\theta})$. We recall that under standard assumptions (see below) we have $\langle\partial_\mu C\rangle = 0$ [35,36]. Hence, from Chebyshev's inequality, Eq. (8) bounds the probability that the cost function partial derivative deviates from zero by a value larger than a given $c \geq 0$ as

$$\Pr[|\partial_\mu C| \geq c] \leq \frac{\text{Var}[\partial_\mu C]}{c^2}. \qquad (9)$$

Hence, if the variance is exponentially small (i.e., if the cost exhibits a barren plateau), the cost function gradient will be on average exponentially small, and an exponential precision will be needed to navigate the flat landscape. In contrast, large variances show that no barren plateaus are present and that the trainability of the parameters under random initialization can be guaranteed.

Let us now present the explicit form of $\partial_\mu C$. From Eqs. (4) and (6), we find that

$$\partial_\mu C = \text{Tr}[W_A V_L \sigma V_L^\dagger W_A^\dagger [H_\mu, W_B^\dagger V_R^\dagger \tilde{O} V_R W_B]], \qquad (10)$$

where we write $V = V_R W V_L$. That is, $V_R$ and $V_L$ contain all gates in the QCNN except for $W$. We henceforth omit the parameter dependence for simplicity of notation. Moreover, from the expansion of $W$ in Eq. (6) we also define

$$W_A = \prod_{\eta \leq \mu} e^{-i\theta_\eta H_\eta} W_\eta, \qquad W_B = \prod_{\eta > \mu} e^{-i\theta_\eta H_\eta} W_\eta, \qquad (11)$$

so that $W = W_B W_A$. We remark that it is straightforward to verify that if $H_\mu$ is an idempotent operator, then $\langle\partial_\mu C\rangle = 0$.

### E. Haar-distributed unitaries

When training the QCNN, a common strategy is to randomly initialize the parameters in $V(\boldsymbol{\theta})$ [34]. Hence, it is convenient to define $\mathcal{W}_{ij}$ as the set of the unitaries obtained from $W_{ij}$ for each random initialization. To analyze the trainability of the QCNN, we assume that the sets $\mathcal{W}_{ij}$ form independent local 2-designs. We outline in Appendix E a method to generate unitaries on two qubits that form

2-designs that do not incur additional depth in the quantum circuit. Moreover, we recall that it has been shown that the more expressible a parametrized quantum circuit is (that is, the more it resembles a 2-design), the harder it is to train it [60]. Hence, studying the case when the unitaries in the convolutional layers form 2-designs constitutes a sort of "most random case" scenario analysis. Thus, if one can show large gradients in this scenario, one can expect that other cases where the unitaries do not form 2-designs should also be trainable, as these unitaries typically have larger gradients than those forming 2-designs [60].

Let us recall the definition of a $t$-design. Let $\mathcal{W} = \{W_y\}_{y\in Y}$ be a set of unitaries in a $d$-dimensional Hilbert space. Then, $\mathcal{W}$ is a $t$-design if for every polynomial $P_{t,t}(W)$ of degree $t$ in the matrix elements of $W$ and of degree $t$ in those of $W^\dagger$, the expectation value of $P_{t,t}(W)$ over the set $\mathcal{W}$ is equal to the expectation value over the Haar distribution. That is,

$$\frac{1}{|Y|}\sum_{y\in Y}P_{t,t}(W_y) = \int d\mu(W)P_{t,t}(W), \qquad (12)$$

where the integral is taken with respect to the Haar measure over the unitary group of degree $d$.

The previous assumption has the following key consequence. Let us recall that the pooling layer is formed by pooling operators defined according to Eq. (7). Then, as shown in Fig. 1 every pooling operator $I$ is preceded by a unitary $W$ in the second sublayer of the convolutional layer such that their combined action can be obtained from $IW$. As $W$ forms a 2-design, it is left and right invariant under the action of the unitary group. That is, for any function $F(W)$, and for any unitary matrix $A$,

$$\int F(AW)d\mu(W) = \int F(W)d\mu(W). \qquad (13)$$

Equation (13) shows that the action of the operators $I_{ij}$ in the pooling layer can be absorbed into the action of the unitaries in the convolutional layers as the variance of the cost function partial derivative is independent of the controlled unitaries in $I_{ij}$. Hence, as schematically shown in Fig. 2, we can consider that the unitary $V(\boldsymbol{\theta})$ of the QCNN is simply composed of two-qubit unitaries $W_{ij}$ which form independent 2-designs.

Finally, let us remark that as shown in Ref. [36], if $W_A$ and $W_B$ form independent 2-designs, then $\langle\partial_\mu C\rangle = 0$ and Eq. (8) can be expressed as

$$\text{Var}[\partial_\mu C] = \frac{2\text{Tr}[H_\mu^2]}{225}\sum_{\substack{pq, \\ p'q'}}\langle\Delta\Omega_{qp}^{q'p'}\rangle_{V_R}\langle\Delta\Psi_{pq}^{p'q'}\rangle_{V_L}. \qquad (14)$$

Here the summation runs over all bit strings $p, q, p'p', q'$ of length $2^{n-2}$. In addition, we define
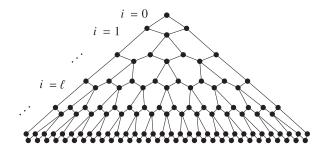
FIG. 2. Tensor network representation of the QCNN. Here each unitary $W_{ij}$ in the convolutional and fully connected layers forms a 2-design. The operators $I_{ij}$ of Eq. (7) in the pooling layer do not modify the variance $\mathrm{Var}[\partial_\mu C]$ as their action can be absorbed by the action of the unitaries in the convolutional layers. Each tensor (circle) corresponds to a $W_{ij}$, with $L = 0$ corresponding to the unitary in the fully connected layer. Circle with three legs indicates that a qubit is traced out after their action.

$$\Delta\Omega_{qp}^{q'p'} = \mathrm{Tr}[\Omega_{qp}\Omega_{q'p'}] - \frac{\mathrm{Tr}[\Omega_{qp}]\mathrm{Tr}[\Omega_{q'p'}]}{4}, \quad (15)$$

$$\Delta\Psi_{pq}^{p'q'} = \mathrm{Tr}[\Psi_{pq}\Psi_{p'q'}] - \frac{\mathrm{Tr}[\Psi_{pq}]\mathrm{Tr}[\Psi_{p'q'}]}{4}, \quad (16)$$

where $\Omega_{qp}$ and $\Psi_{qp}$ are operators on $\mathcal{H}_w$ defined as

$$\Omega_{qp} = \mathrm{Tr}_{\bar{w}}[(|p\rangle\langle q| \otimes \mathbb{1}_w)V_R^\dagger \tilde{O} V_R], \quad (17)$$

$$\Psi_{pq} = \mathrm{Tr}_{\bar{w}}[(|q\rangle\langle p| \otimes \mathbb{1}_w)V_L \sigma V_L^\dagger]. \quad (18)$$

We use $\mathrm{Tr}_{\bar{w}}$ to denote the trace over all qubits not in $\mathcal{H}_w$.

## III. MAIN RESULTS

In this section, we present our main results. First, we introduce a novel method for analyzing the scaling of the variance of Eq. (14) which we call the GRIM. Specifically, our method can be used when employing the Weingarten calculus to integrate unitaries over the unitary group. As we discuss below, the GRIM is based on finding recursions when integrating groups of unitaries to form a graph which can be readily used to compute the scaling of the cost function partial derivative variance in Eq. (14). Our second main result is that we employ the GRIM to obtain a lower bound on $\mathrm{Var}[\partial_\mu C]$ for the QCNN architecture. Moreover, this lower bound can be used, under certain standard assumptions, to guarantee the trainability of the QCNN. We then present our results to guarantee the trainability of a pooling-based QCNN. Finally, we present heuristic results obtained from numerically computing $\mathrm{Var}[\partial_\mu C]$.

### A. Graph recursion integration method

Let us consider the task of computing the expectation values $\langle\Delta\Omega_{qp}^{q'p'}\rangle_{V_R}$ and $\langle\Delta\Psi_{pq}^{p'q'}\rangle_{V_L}$ in Eq. (14). In Ref. [36], it was shown that one can solve this problem by sequentially

integrating each Haar-distributed unitary via the elementwise formula of the Weingarten calculus [61,62] (see the Appendix).

While the previous approach has been successfully used to analyze the trainability of parametrized quantum circuits and of QNNs [36,37], it has the difficulty that the number of terms that one has to keep track of increases exponentially with the number of unitaries integrated. The GRIM allows us to circumvent this difficulty by simultaneously integrating specific groups of unitaries and recursively grouping the resulting terms to form a *graph* $\mathcal{G}_w$. As such, the GRIM has to be implemented to form a graph for the integration of the gates in $V_L$ and likewise for the integration of the gates in $V_R$. In what follows, we consider the case of obtaining the graph to compute the expectation value of $\langle\Delta\Omega_{qp}^{q'p'}\rangle_{V_R}$. Moreover, we henceforth assume that $V_R$ is composed of all the unitaries in the forward light cone $\mathcal{L}_F$ of $W$.

The first step of the GRIM is to group the unitaries in $V_R$ into *modules* that cover all of $\mathcal{L}_F$. We remark that different modules will lead to different graphs and that the fewer distinct modules one employs, the smaller the graph will be in terms of the number of distinct nodes. In Fig. 3(a), we show the three distinct basic modules needed to analyze the QCNN architecture, such that for any $W$ in the $\ell$th layer, $\mathcal{L}_F$ can be covered with $\ell$ such modules. Specifically, Fig. 3(b) depicts an example where $\mathcal{L}_F$ can be covered by three middle modules $M_\mathcal{M}$, an edge module $M_\mathcal{E}$, and a center module $M_\mathcal{C}$.

The next step in the GRIM is to integrate the gates in each module $M$ to form the graph. Each node $N_i \in \mathcal{G}_w$ will correspond to different contractions of operators containing
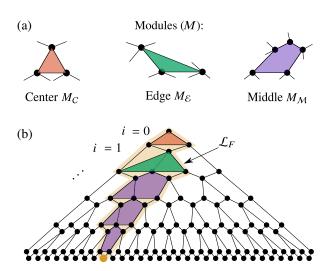


FIG. 3. GRIM modules for the QCNN architecture. (a) Basic modules $M$ from left to right; we refer to these modules as the center $M_\mathcal{C}$, edge $M_\mathcal{E}$, and middle $M_\mathcal{M}$ module. Note that while both $M_\mathcal{C}$ and $M_\mathcal{E}$ contain three unitaries, their contractions are different. (b) Example of the first step of the GRIM. The forward light cone $\mathcal{L}_F$ of $W$ can be covered with the three basic modules. Here, $W$ is indicated with a larger colored circle.

the modules, while the edges of the graph will be associated with real coefficients $\lambda_{i,j} \in (0, 1)$. In this notation, the coefficient $\lambda_{i,j}$ corresponds to the oriented edge $N_i \to N_j$. Moreover, the initial node can always be defined from Eq. (15) as

$$N_1(T) = \text{Tr}[TT'] - \frac{\text{Tr}[T]\text{Tr}[T']}{4}, \qquad (19)$$

where $T$ and $T'$ are two-qubit operators. Following Eq. (17), here $T'$ contains primed operators obtained from $|\boldsymbol{p}'\rangle\langle\boldsymbol{q}'|$. Then, as shown in the Appendix, the result of integrating a given module $M$ can be expressed as

$$\int_M \prod d\mu(W_{ij})N_i(T) = \sum_j e_{ij}N_j(\tilde{T}), \qquad (20)$$

where the integral is taken over the unitaries $W_{ij}$ that appear in a module $M$, and where the terms $e_{ij}$ lead to the edge coefficients $\lambda_{i,j}$. Moreover, here the operator $\tilde{T}$ is obtained from $T$ such that it contains the same modules as the ones in $T$ except for the one that was integrated according to Eq. (20).

Equation (20) is at the basis of the GRIM as it shows that when a module is integrated, one connects the nodes $N_i \to N_j$ for all $j$ in the summation. By sequentially integrating all unitaries, one can form an oriented graph which always starts with $N_1$ according to Eq. (15). As we discuss in the Appendix, the number of nodes in the graph, denoted as $|\mathcal{G}_w|$, is always in $\mathcal{O}(\ell)$ for the QCNN architecture. Additionally, the structure of the graphs $\mathcal{G}_w$ is recursive, which allows us to obtain results that are valid for arbitrary $\ell$ and for all placements of the unitary $W$. In Fig. 4(a), we show an example where $W$ is in the edge of the $\ell$th layer and where the forward light cone $\mathcal{L}_F$ is composed of $(\ell - 1)$ edge modules and a single center module. The graph that arises from integrating the unitaries in the $M_\mathcal{E}$ modules is shown in Fig. 4(b). We refer the reader to the Appendix for additional details on the derivation of this graph and the definition of the $N_i$ nodes. For this specific case, the graph contains $|\mathcal{G}_w| = (\ell + 1)$ nodes.

Once the graph $\mathcal{G}_w$ has been formed, one can use it to explicitly compute the contribution of expectation values in the variance of Eq. (14). Specifically, the following proposition proved in the Appendix holds.

**Proposition 1.** Let $W$ be a unitary in the $\ell$th layer of the QCNN. The contribution of $\langle\Delta\Omega_{qp}^{q'p'}\rangle_{V_R}$ to the variance can be computed via the GRIM as

$$\text{Var}[\partial_\mu C] \geq \frac{\text{Tr}[H_\mu^2]\varepsilon_O}{450}\langle\varepsilon_{\tilde{\sigma}_w}\rangle_{V_L}\sum_{\boldsymbol{g}\in P_\ell(\mathcal{G}_w)}\Lambda_{\boldsymbol{g}}, \qquad (21)$$

where $\tilde{\sigma}_w = \text{Tr}_{\bar{w}}[V_L\sigma V_L^\dagger]$ is the reduced operator $\sigma$ in the subsystem of the qubits that $W$ acts on, and where we defined for an operator $O$
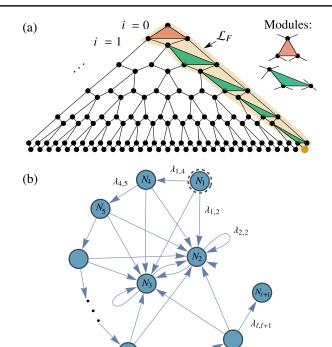


FIG. 4. GRIM for the case when $W$ is in the edge of $\ell$th layer of the QCNN. (a) The forward light cone can be covered with $(\ell - 1)$ edge modules $M_\mathcal{E}$ and a single center module $M_\mathcal{C}$. Here, $W$ is indicated with a larger colored circle. (b) Graph obtained from the GRIM by integrating all modules $M_\mathcal{E}$ in ar$\mathcal{L}_F$ for the case in (a). Graph $\mathcal{G}_w$ obtained via GRIM. Here the edge coefficients $\lambda_{i,j}$ are real and positive numbers such that $\lambda_{i,j} \in (0, 1)$. Moreover, for this case the number of nodes is $(\ell + 1)$, and the structure of the graph is such that one can easily obtain the graph for any value of $\ell$.

$$\varepsilon_O = D_{\text{HS}}\left(O, \text{Tr}[O]\frac{\mathbb{1}}{4}\right), \qquad (22)$$

with $D_{\text{HS}}(A, B) = \text{Tr}[(A - B)^2]$ the Hilbert-Schmidt distance between $A$ and $B$. Here $\mathcal{G}_w$ denotes the oriented graph obtained by integrating all the modules in $\mathcal{L}_F$, $P_\ell(\mathcal{G}_w)$ is the set of all paths of length $\ell$ over $\mathcal{G}_w$, and $\boldsymbol{g} = (g_1, \dots, g_\ell)$ is a vector which indicates the nodes in the path such that $g_i \in [1, \dots, |\mathcal{G}_w|]$. Finally, $\Lambda_{\boldsymbol{g}}$ are real positive numbers obtained as

$$\Lambda_{\boldsymbol{g}} = \prod_{\xi=1}^{\ell} \lambda_{g_\xi, g_{\xi+1}}, \qquad (23)$$

where $\lambda_{g_\xi, g_{\xi+1}} \in (0, 1)$ are the edge coefficients associated with the oriented edge $N_\xi \to N_{\xi+1}$.

Note that $\varepsilon_O = N_1(O)$ from Eq. (19) by taking $T = T'$, meaning that here we are using the surprising fact that the final node is *always* $N_1(O)$. Proposition 1 shows that given a unitary $W$, the contribution from the expectation

value of $\Delta\Omega_{pq}^{p'q'}$ can be ultimately obtained by adding up all the paths in the oriented graph $\mathcal{G}_w$ weighed by the coefficients associated with the edges taken over the path. Moreover, since $\varepsilon_O$ and $\varepsilon_{\tilde{\sigma}_w}$ are always positive, any single path over the graph leads to a lower bound for $\mathrm{Var}[\partial_\mu C]$. Finally, we remark that one can also employ the GRIM to integrate the gates in $V_L$ to compute the expectation value $\langle \varepsilon_{\tilde{\sigma}_w} \rangle_{V_L}$.

In what follows, we briefly give an example of how to use the GRIM to compute the expectation value $\langle \Delta\Omega_{qp}^{q'p'} \rangle_{V_R}$ for the case when $W$ is the center unitary of the $\ell$th layer of the QCNN as indicated in Fig. 5(a). Here we can see that $\mathcal{L}_F$ can be covered with $\ell$ center modules $M_C$. Employing the notation $\tilde{O}_\ell = \Omega_{qp}$, it is straightforward to see that $\langle \Delta\Omega_{qp}^{q'p'} \rangle_{V_R} = \langle N_1(\tilde{O}_\ell) \rangle_{V_R}$. Then, as shown in Fig. 5(b) in the tensor network representation of quantum circuits, we can always write the recursion relation

$$\tilde{O}_\ell = \mathrm{Tr}_{\bar{w}}[(|\boldsymbol{p}\rangle\langle\boldsymbol{q}|_{\ell-1} \otimes \mathbb{1}_w) M_C \tilde{O}_{\ell-1} M_C^\dagger]. \quad (24)$$



(a)

Module:

(b) $N_1(\bigcirc\!\!-\!\!\bigcirc) = \bigcirc\!\!\bigcirc\!\!\bigcirc - \frac{1}{4}\bigcirc\!\!\bigcirc$

$\bigcirc\!\!-\!\!\!\fbox{$\ell+1$}\!\!-\!\!\bigcirc = $ ... $\bigcirc\!\!-\!\!\ell\!\!-\!\!\bigcirc = \tilde{O}_\ell$

$\fbox{}-\fbox{} = |p\rangle\langle q|_\ell$

(c) $\int_{M_C}\prod d\mu(W_{ij})N_1(\bigcirc\!\!-\!\!\fbox{$\ell+1$}\!\!-\!\!\bigcirc) = N_1(\bigcirc\!\!-\!\!\ell\!\!-\!\!\bigcirc)\left[\frac{4}{25}\underset{p_1}{\bigcirc\!\!\ell\!\!\bigcirc} + \frac{8}{125}\underset{p_2\; p_3}{\left(\!\!\begin{array}{c}\ell\\\ell\end{array}\!\! + \!\!\begin{array}{c}\ell\\\ell\end{array}\!\!\right)}\right]$
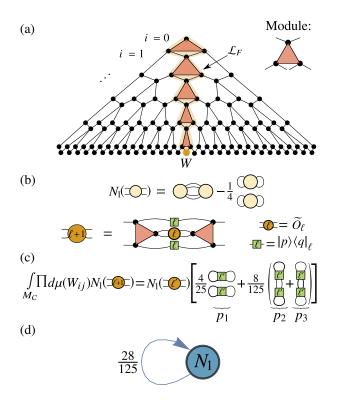
(d) $\frac{28}{125}\;\;\bigcirc N_1$

FIG. 5. GRIM for the case when $W$ is in the center of $\ell$th layer of the QCNN. (a) The unitaries in $\mathcal{L}_F$ can be covered by $\ell$ center modules $M_C$. (b) Schematic representation of the tensor contraction $N_1$ of Eq. (19). Here we also show the recursive relation between the operators $\tilde{O}_\ell$. (c) Tensor network representation of Eq. (25) obtained by integrating the $(\ell + 1)$th center module. The operators $p_2$ and $p_3$ correspond to contractions over different qubits, as indicated by the bold lines. (d) Graph $\mathcal{G}_w$ obtained from the GRIM for the case in (a).

Here, $|\boldsymbol{p}\rangle\langle\boldsymbol{q}|_{\ell-1}$ denotes the projectors on the qubits that the $\ell$th module $M_C$ acts on. With this notation. it then follows that $\tilde{O}_0 = O$.

By employing the elementwise formula of the Weingarten calculus to integrate unitaries over the Haar distribution (see the Appendix), we find for all $\ell' \leq \ell$,

$$\int_{M_C} N_1(\tilde{O}_{\ell'})d\mu = \frac{4N_1(\tilde{O}_{\ell'-1})}{25}\left(p_1 + \frac{2(p_2+p_3)}{5}\right), \quad (25)$$

where the integration is over all unitaries $W_{ij}$ in the $\ell'$th module $M_C$. As depicted in Fig. 5(c), the coefficients $p_i$ with $i = 1, 2, 3$ arise from different contractions of $|\boldsymbol{p}\rangle\langle\boldsymbol{q}|_{\ell'-1}$. Then, as shown in the Appendix, the operators $p_2$ and $p_3$ lead to a factor $1/2$, while the operator $p_1$ leads to a factor of 1 so that $\frac{4}{25}\{e_1 + [2(e_2 + e_3)/5]\} = 28/125$. Equation (25) shows how the structure of the graph $\mathcal{G}_w$ emerges, as integrating a module connects the node $N_1$ with itself with a coefficient $\lambda_{1,1} = 28/125$. The ensuing graph is presented in Fig. 5(d), and we can see that it is composed of a single node.

## B. Trainability of the QCNN

Here we present the main result of our article, which is derived using the GRIM. The detailed proofs can be found in the Appendix. For convenience of notation, here we consider the case where $W$ is in the first sublayer. The case where it belongs to the second sublayer is considered in the Appendix, and we remark that the results are substantially the same for both cases, albeit with some notational differences.

**Theorem 1.** Consider the QCNN cost function of Eq. (4) and let $W$ be a unitary in the first sublayer of the $\ell$th layer of $V(\boldsymbol{\theta})$. Moreover, let us assume that the unitaries $W_{ij}$, $W_A$, and $W_B$ in the convolutional and fully connected layers form independent 2-designs. Then, the lower bound of the variance of the cost partial derivative with respect to a parameter $\theta_\mu$ in unitary $W$ can be computed using the GRIM from the graph $\mathcal{G}_w$ as

$$\mathrm{Var}[\partial_\mu C] \geq F_n(L, \ell), \quad (26)$$

with

$$F_n(L, \ell) = \frac{1}{9}\frac{\mathrm{Tr}[H_\mu^2]\varepsilon_O\varepsilon_{\sigma_w}}{50^{L-\ell+1}}\sum_{g \in P_\ell(\mathcal{G}_w)}\Lambda_g. \quad (27)$$

Here, $\sigma_w = \mathrm{Tr}_{\bar{w}}[\sigma]$ is the reduced operator $\sigma$ in the subsystem of the qubits that $W$ acts on. Moreover, $\varepsilon_O$, $P_\ell(\mathcal{G}_w)$, $g$, and $\Lambda_g$ are defined in Proposition 1.

From Theorem 1, we can obtain the following corollary.

**Corollary 1.** Consider the function $F_n(L, \ell)$ defined in Eq. (27). Then, since $L$ is at most $\mathcal{O}(\log(n))$, the variance $\mathrm{Var}[\partial_\mu C]$ is at most polynomially vanishing with $n$ as

$$F_n(L, \ell) \in \Omega\left(\frac{1}{\text{poly}(n)}\right), \tag{28}$$

provided that $\text{Tr}[H_\mu^2]\varepsilon_O\varepsilon_{\sigma_w} \in \Omega[1/\text{poly}(n)]$.

Let us now consider the main implication of this result. Corollary 1 provides conditions under which no barren plateaus arise, and the trainability of the parameters in the unitaries of the QCNN architecture can be guaranteed. Indeed, the determination of a cost-minimizing direction in the parameter hyperspace requires only a polynomially large precision. This is in contrast to landscapes which exhibit a barren plateau, where an exponentially large precision is needed to navigate through the flat landscape.

We remark that at the core of the result in Corollary 1 lies the fact that QCNNs are shallow, as they have at most a number of layers $L$ in $\mathcal{O}(\log(n))$. Such shallowness naturally arises from the intrinsic structure of the QCNN architecture, where the number of qubits is reduced at each layer. Hence, terms such as $1/50^{L-\ell+1}$ or $\Lambda_g$ in Theorem 1 can vanish no faster than $\Omega(1/\text{poly}(n))$ if $L$ is in $\mathcal{O}(\log(n))$.

As indicated in Corollary 1, the bound in Eq. (28) holds if $\text{Tr}[H_\mu^2]\varepsilon_O\varepsilon_{\sigma_w} \in \Omega(1/\text{poly}(n))$. This assumption means that $O$ and the reduced operator $\sigma_w$ need to, respectively, have a Hilbert-Schmidt distance to the (scaled) identity operator which is not exponentially vanishing. Note that this condition is expected, as extracting information by measuring operators close to the identity is naturally a hard task. Similarly, attempting to train a quantum circuit on a state close to the identity is also a difficult task. Here we remark that in the Appendix we relax the condition on $\sigma$ so that we can guarantee that the lower bound for $\text{Var}[\partial_\mu C]$ is at most polynomially vanishing with $n$ if $\sigma_z$ is not exponentially close to $\text{Tr}[\sigma]\mathbb{1}/4$ (as measured by the contraction in $N_1$) on *any* pair of qubits in the backward-propagated light cone of $W$.

### C. Trainability from pooling

As previously discussed, when the unitaries $W_{ij}$ in the convolutional layers form 2-designs, the action of the operators $I_{ij}$ in the pooling modules can be absorbed by the action of their neighboring $W_{ij}$. Here, we analyze the trainability of the QCNN for the special case when the unitaries $W_{ij}$ act trivially, i.e., when $W_{ij} = \mathbb{1} \; \forall \; i, j$. This case can be illustrated by the pooling module in Fig. 6, in which an input state $(\rho^{(0)})^{\otimes n}$ is processed by pooling layers locally described by the channel

$$\rho^{(i+1)} = \text{tr}_A[I_{i+1}(\rho^{(i)})^{\otimes 2}I_{i+1}^\dagger], \tag{29}$$

where

$$I_{i+1} = |+\rangle\langle+|_A \otimes e^{-i\theta_+^{(i+1)}Y} + |-\rangle\langle-|_A \otimes e^{-i\theta_-^{(i+1)}Y} \tag{30}$$



$$\sigma^{(j)} = \rho^{(j)}(\{\theta_\pm^{(k)}\}_{k=1}^j) \qquad \sigma^{(j+1)} = \rho^{(j+1)}(\{\theta_\pm^{(k)}\}_{k=1}^j, \theta_\pm^{(j+1)})$$
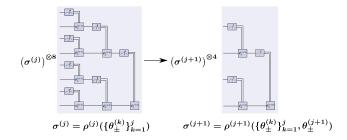
FIG. 6. Schematic circuits of a pooling-based QCNN. The left module belongs to the layer that acts prior to one which contains the right module. As shown, after each pooling layer, half the qubits are measured. In this special case, the unitaries in the convolutional and fully connected layers of the QCNN act trivially.

is the controlled unitary corresponding to the pooling. Moreover, here we assume for simplicity that $\rho^{(0)} = |0\rangle\langle0|^{\otimes n}$, where we recall that $n = 2^L$.

As explicitly proved in the Appendix, the following theorem holds.

**Theorem 2.** Consider a QCNN where the convolutional unitaries are $W_{ij} = \mathbb{1}$ for all $i$, $j$ and where the action of pooling layers can be described by Eqs. (29) and (30). Then, the expectation value of the magnitude of the cost function partial derivative $\partial C/\partial \theta^{(k)} = \partial_k C$ with respect to any parameter $\theta^{(k)}$ in a pooling module is

$$\langle|\partial_k C|\rangle = \frac{1}{2n^{\log_2(\pi)-1}}. \tag{31}$$

Theorem 2 shows that the pooling-based QCNN does not exhibit a barren plateau as $\langle|\partial_k C|\rangle$ is polynomially vanishing with the system size.
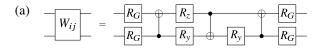
### D. Numerical verification

In this section, we present numerical results to analyze the scaling of the cost function partial derivative variance. Specifically, we consider here two cases: when all the unitaries $W_{ij}$ in each convolutional module are independent, and when all the unitaries in the same layer are identical. The second case corresponds to the QCNN architecture as originally introduced in Ref. [34]. As we show below, our numerics indicate that correlating ("corr") the unitaries $W_{ij}$ (to make them identical) leads to a variance of the cost function partial derivatives which is larger to the one obtained in the uncorrelated ("uncorr") case, i.e.,

$$\text{Var}[\partial_\mu C_{\text{corr}}] \geq \text{Var}[\partial_\mu C_{\text{uncorr}}]. \tag{32}$$

This numerical result suggests that the lower bound obtained in Theorem 1 can also be used as a lower bound for the correlated case. We remark that in Ref. [47] it was shown that correlating the parameters in an ansatz can lead to larger variances.

The QCNN in our heuristics is constructed such that each $W_{ij}$ is given by the parametrized circuit of Fig. 7(a). As shown in Ref. [63], this decomposition allows us to prepare any two-qubit unitary (up to a global phase). Since we simulate QCNNs with an even number of qubits, which is not always a power of 2, in the pooling layers we trace out qubits so that each convolutional layer always acts on an even number of qubits. Moreover, for simplicity, we consider the case where the training set contains a single state initialized by the all-zero state $\rho = |0\rangle\langle 0|^{\otimes n}$, and at the end of the QCNN, we measure the operator $O = Z \otimes Z$, where $Z$ denotes the Pauli-$z$ operator. Finally, here the trainable parameter $\theta_\mu$ is in the center unitary acting on the first layer. We remark that the results obtained for this case are representative of other possible choices for the location of $\theta_\mu$.

For each number of qubits $n \in \{4, 6, \ldots, 26\}$, we simulate 200 instances of randomly initialized QCNNs and compute the derivative $\partial_\mu C$. All the simulations are performed using TENSORFLOW QUANTUM [64]. In Fig. 7(b), we show the results for $\mathrm{Var}[\partial_\mu C]$ as a function of $n$ for the cases when the $W_{ij}$ are correlated and uncorrelated. First, let us remark that the variance for the correlated case is always larger than that of the uncorrelated case as in Eq. (32). Moreover, noting that
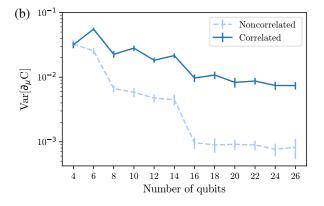


FIG. 7. Ansatz and results for the numerical simulations of the QCNN. (a) Ansatz for the two-qubit unitary $W_{ij}$. Here $R_G = e^{-iZ\theta_1/2}e^{-iY\theta_2/2}e^{-iZ\theta_3/2}$ is a general single-qubit rotation parametrized by three angles, and $X$, $Y$ are Pauli operators. In addition, $R_y$ and $R_z$ denote single-qubit rotations about the $y$ and $z$ axes, respectively. (b) Heuristic scaling of the variance of the cost function partial derivative $\mathrm{Var}[\partial_\mu C]$. The $y$ axis is in a log scale. The solid (dashed) curve represents results for correlated (uncorrelated) unitaries in the convolutional layers. Error bars represent the standard deviation over 16 repetitions. Here we can see that $\mathrm{Var}[\partial_\mu C]$ is always larger when the unitaries are correlated. As $n$ increases, we see that the curves are sublinear, indicating a subexponential scaling for $\mathrm{Var}[\partial_\mu C]$.

the $y$ axis is a log scale, we can see that the curves are sublinear, meaning that the scaling of $\mathrm{Var}[\partial_\mu C]$ is subexponential. This result then numerically verifies that QCNNs do not exhibit barren plateaus in their training landscape.

## IV. DISCUSSION

Investigating the barren plateau phenomenon in parametrized quantum circuits and quantum neural networks is fundamental to understanding their trainability. In the absence of a barren plateau, the determination of a minimizing direction in the cost function landscape does not require an exponentially large precision, meaning that one can always navigate through the landscape by measuring expectation values with a precision that grows at most polynomially with the system size. We remark that polynomial overhead is the standard goal for quantum algorithms, which aim to achieve a speed-up over classical algorithms that often scale exponentially. Hence, the absence of a barren plateau is a necessary (although not sufficient) condition to determine whether or not a quantum speed-up is achievable.

In this work, we analyze the gradient scaling for the QCNN architecture. We first introduce a novel method called the GRIM, which greatly simplifies the computation of expectation values of quantities that are expressed in terms of Haar-distributed unitaries. The goal of the GRIM is to create an oriented graph that can be used to evaluate this expectation value.

We then employ the GRIM to derive our main result in Theorem 1, which provides a general lower bound on the variance of the cost function gradient. In Corollary 1, we derive the asymptotic scaling of this lower bound under standard assumptions and show that it vanishes no faster than polynomially with the system size. This result confirms the absence of barren plateaus in QCNNs and allows us to guarantee the trainability of this architecture when randomly initializing its parameters. Moreover, our work highlights QCNNs as potentially being generically trainable under random initialization, and hence sets them apart from other QNN architectures that are trainable only under certain conditions.

Since the results in Theorem 1 are derived assuming that the two-qubit unitaries in the convolutional layers form independent 2-designs, we additionally consider the case when they are trivial and equal to the identity. In Theorem 2, we show that this pooling-based QCNN, i.e., a QCNN which is composed only of pooling layers, is also trainable when initialized randomly. In addition, we present numerical evidence to support our theoretical results, and we show that correlating the parameters in the convolutional layers (as in the original QCNN architecture) leads to larger gradient variances, meaning that our results can also be useful for that particular case. Similarly, when the two-qubit gates do not form 2-designs, then one can expect the variance of the gradient to be larger [60]. Moreover, let us remark that while

we consider cost functions which are linear with respect to the input density matrices, as shown in Sec. II B, this type of cost is quite general and encompasses a wide range of applications (such as classification).

While our work proves the trainability for the QCNN architecture, there are many future research directions that can be pursued to generalize our results. First, one can analyze more general cost functions than the ones considered here. For instance, one could consider mean-square cost functions, as the ones employed in regression problems, where $C$ is a quadratic function on the expectation value $\mathrm{Tr}[V(\boldsymbol{\theta})\sigma C^\dagger(\boldsymbol{\theta})O]$. In addition, due to close connection between the QCNN architecture and the multiscale entanglement renormalization ansatz (MERA) [34], it would be interesting to analyze how our results and methods can be employed to derive trainability results for MERA frameworks. Finally, we believe the GRIM will be useful in analyzing the scaling of the cost function partial derivative variance for other VQA and QNN architectures.

*Note added.*—Recently, Ref. [65] was posted where the authors analyze the existence of barren plateaus in QNN with a tree tensor structure, concluding that this architecture also does not exhibit barren plateaus.

## APPENDIX A: PRELIMINARIES

In this Appendix, we present the notations and preliminary results needed to derive our main theorems.

### 1. Haar-distributed unitaries and integration over the unitary group

Here, let us recall the definition of a $t$-design. Let $\mathcal{W} = \{W_y\}_{y \in Y}$ be a finite set of size $|Y|$ of $d$-dimensional unitaries $W_y$, and let $P_{t,t}(W)$ be a polynomial of degree at most $t$ in the matrix elements of $W$, and at most of degree

$t$ in those of $W^\dagger$. We say that $\mathcal{W}$ is a $t$-design if for every polynomial $P_{t,t}(W)$ we have

$$\frac{1}{|Y|}\sum_{y \in Y}P_{t,t}(W_y) = \int d\mu(W)P_{t,t}(W), \qquad \text{(A1)}$$

where the integral is taken with respect to the Haar measure over the unitary group.

Then, to compute expectation values as in Eq. (A1), it is useful to recall formulas which allow for symbolic integration with respect to the Haar measure on the unitary group $\mathcal{U}(d)$ of degree $d$. Specifically, following the elementwise formula of the Weingarten calculus [61,62], we have that the following formulas are valid for the first two moments:

$$\int d\mu(W)w_{i_1j_1}w^*_{i_2j_2} = \frac{\delta_{i_1i_2}\delta_{j_1j_2}}{d},$$

$$\int d\mu(W)w_{i_1j_1}w_{i_2j_2}w^*_{i_1'j_1'}w^*_{i_2'j_2'} = \frac{1}{d^2-1}\left(\Delta_1 - \frac{\Delta_2}{d}\right), \quad \text{(A2)}$$

where $w_{ij}$ are the matrix elements of the unitary $W \in \mathcal{U}(d)$, and

$$\Delta_1 = \delta_{i_1i_1'}\delta_{i_2i_2'}\delta_{j_1j_1'}\delta_{j_2j_2'} + \delta_{i_1i_2'}\delta_{i_2i_1'}\delta_{j_1j_2'}\delta_{j_2j_1'},$$

$$\Delta_2 = \delta_{i_1i_1'}\delta_{i_2i_2'}\delta_{j_1j_2'}\delta_{j_2j_1'} + \delta_{i_1i_2'}\delta_{i_2i_1'}\delta_{j_1j_1'}\delta_{j_2j_2'}. \quad \text{(A3)}$$

Here we remark that the integrations in our calculations are performed by employing the Random Tensor Network Integrator package for symbolic integration of Haar-distributed tensors [66].

### 2. Useful results

**Lemma 1.** Let $\mathcal{H} = \mathcal{H}_1 \otimes \mathcal{H}_2 \otimes \mathcal{H}_3$ be a tripartite Hilbert space, and let $\{|p\rangle\}$ be an arbitrary basis of $\mathcal{H}_3$. Then for any pair of linear operators $A, B : \mathcal{H} \to \mathcal{H}$, the following identities hold:

$$\sum_{p,q}\mathrm{Tr}_2\{\mathrm{Tr}_{13}[(\mathbb{1}_{12} \otimes |p\rangle\langle q|_3)A]\mathrm{Tr}_{13}[(\mathbb{1}_{12} \otimes |q\rangle\langle p|_3)B]\}$$

$$= \mathrm{Tr}_{23}\{\mathrm{Tr}_1[A]\mathrm{Tr}_1[B]\}, \qquad \text{(A4)}$$

$$\sum_{p,q}\mathrm{Tr}_{123}[(\mathbb{1}_{12} \otimes |p\rangle\langle q|_3)A]\mathrm{Tr}_{123}[(\mathbb{1}_{12} \otimes |q\rangle\langle p|_3)B]$$

$$= \mathrm{Tr}_3\{\mathrm{Tr}_{12}[A]\mathrm{Tr}_{12}[B]\}. \qquad \text{(A5)}$$

For the proof of Lemma 1, see Ref. [36].

**Lemma 2.** Let $\mathcal{H} = \mathcal{H}_1 \otimes \mathcal{H}_2$ be a bipartite Hilbert space with $\dim \mathcal{H}_2 = d$, and let $H$ be a Hermitian operator on $\mathcal{H}$, we have

$$\mathrm{Tr}[H^2] - \mathrm{Tr}\left[\frac{\mathrm{Tr}_2[H]^2}{d}\right] = D_{\mathrm{HS}}\left(H, \mathrm{Tr}_2[H] \otimes \frac{\mathbb{1}}{d}\right), \qquad \text{(A6)}$$

where $D_{\mathrm{HS}}(H_1, H_2) = \mathrm{Tr}[(H_1 - H_2)^2] = \|H_1 - H_2\|_2^2$ is the Hilbert-Schmidt distance. In particular, if $\mathcal{H} = \mathcal{H}_2$,

$$\mathrm{Tr}[H^2] - \frac{1}{d}\mathrm{Tr}[H]^2 = D_{\mathrm{HS}}\left(H, \frac{\mathbb{1}}{d}\mathrm{Tr}[H]\right). \tag{A7}$$

*Proof.*—Consider the following chain of equalities:

$$D_{\mathrm{HS}}\left(H, \mathrm{Tr}_2[H] \otimes \frac{\mathbb{1}}{d}\right) = \mathrm{Tr}\left[\left(H - \mathrm{Tr}_2[H] \otimes \frac{\mathbb{1}}{d}\right)^2\right] \tag{A8}$$

$$= \mathrm{Tr}[H^2] + \mathrm{Tr}\left[\mathrm{Tr}_2[H]^2 \otimes \frac{\mathbb{1}}{d^2}\right] - 2\mathrm{Tr}\left[H\left(\mathrm{Tr}_2[H] \otimes \frac{\mathbb{1}}{d}\right)\right] \tag{A9}$$

$$= \mathrm{Tr}[H^2] + \frac{1}{d}\mathrm{Tr}\{\mathrm{Tr}_2[H]^2\} - 2\mathrm{Tr}\left[H\left(\mathrm{Tr}_2[H] \otimes \frac{\mathbb{1}}{d}\right)\right] \tag{A10}$$

$$= \mathrm{Tr}[H^2] + \frac{1}{d}\mathrm{Tr}\{\mathrm{Tr}_2[H]^2\} - \frac{2}{d}\mathrm{Tr}\{\mathrm{Tr}_2[H]^2\} \tag{A11}$$

$$= \mathrm{Tr}[H^2] - \frac{\mathrm{Tr}\{\mathrm{Tr}_2[H]^2\}}{d}. \tag{A12}$$

∎

**Lemma 3.** Consider the operator $\Delta\Psi_{pq}^{p'q'}$ defined in the main text, which we recall here for convenience:

$$\Delta\Psi_{pq}^{p'q'} = \mathrm{Tr}[\Psi_{pq}\Psi_{p'q'}] - \frac{\mathrm{Tr}[\Psi_{pq}]\mathrm{Tr}[\Psi_{p'q'}]}{d}, \tag{A13}$$

where $p$, $q$, $p'$, $q'$ are vectors of length $2^n - d$. Let $\mathcal{H}$ be a Hilbert space of $n$ qubits and let $\mathcal{H}_w$ be a $d$-dimensional subsystem, such that $\mathcal{H}_w \otimes \mathcal{H}_{\bar{w}} = \mathcal{H}$. Then, let $S$ and $\bar{S}$ be two disjoint sets of qubits not in $\mathcal{H}_w$ such that $\mathcal{H}_S \otimes \mathcal{H}_{\bar{S}} = \mathcal{H}_{\bar{w}}$. The following equality holds:

$$\sum_{\substack{pq \\ p'q'}}(\delta_{pq'})_{\bar{S}}(\delta_{qp'})_{\bar{S}}(\delta_{pq})_S(\delta_{p'q'})_S\Delta\Psi_{pq}^{p'q'} = D_{\mathrm{HS}}\left(\tilde{\sigma}_{\bar{S},w}, \tilde{\sigma}_{\bar{S}} \otimes \frac{\mathbb{1}}{d}\right), \tag{A14}$$

where $\tilde{\sigma}_{\bar{S},w} = \mathrm{Tr}_S[\sigma_L]$ and $\sigma_L = V_L\sigma V_L^{\dagger}$. Here, $\mathrm{Tr}_S[\cdot]$ indicates the partial trace over $\mathcal{H}_S$.

*Proof.*—Let us consider the first term in Eq. (A13). We have

$$\sum_{\substack{pq \\ p'q'}}(\delta_{pq'})_{\bar{S}}(\delta_{qp'})_{\bar{S}}(\delta_{pq})_S(\delta_{p'q'})_S\mathrm{Tr}[\Psi_{pq}\Psi_{p'q'}] \tag{A15}$$

$$= \sum_{\substack{pq \\ p'q'}}(\delta_{pq'})_{\bar{S}}(\delta_{qp'})_{\bar{S}}(\delta_{pq})_S(\delta_{p'q'})_S\mathrm{Tr}\{\mathrm{Tr}_{\bar{w}}[(|p\rangle\langle q|_{\bar{S}} \otimes |p\rangle\langle q|_S \otimes \mathbb{1}_d)\sigma_L]\mathrm{Tr}_{\bar{w}}[(|p'\rangle\langle q'|_{\bar{S}} \otimes |p'\rangle\langle q'|_S \otimes \mathbb{1}_w)\sigma_L]\} \tag{A16}$$

$$= \sum_{pq\in\bar{S}}\sum_{pp'\in S}\mathrm{Tr}\{\mathrm{Tr}_{\bar{w}}[(|p\rangle\langle q|_{\bar{S}} \otimes |p\rangle\langle p|_S \otimes \mathbb{1}_w)\sigma_L]\mathrm{Tr}_{\bar{w}}[(|q\rangle\langle p|_{\bar{S}} \otimes |p'\rangle\langle p'|_S \otimes \mathbb{1}_w)\sigma_L]\} \tag{A17}$$

$$= \sum_{pq\in\bar{S}}\mathrm{Tr}\left\{\mathrm{Tr}_{\bar{w}}\left[\left(|p\rangle\langle q|_{\bar{S}} \otimes \sum_p |p\rangle\langle p|_S \otimes \mathbb{1}_w\right)\sigma_L\right]\mathrm{Tr}_{\bar{w}}\left[\left(|q\rangle\langle p|_{\bar{S}} \otimes \sum_{p'}|p'\rangle\langle p'|_S \otimes \mathbb{1}_w\right)\sigma_L\right]\right\} \tag{A18}$$

$$= \sum_{pq\in\bar{S}}\mathrm{Tr}\{\mathrm{Tr}_{\bar{w}}[(|p\rangle\langle q|_{\bar{S}} \otimes \mathbb{1}_{w,S})\sigma_L]\mathrm{Tr}_{\bar{w}}[(|q\rangle\langle p|_{\bar{S}} \otimes \mathbb{1}_{w,S})\sigma_L]\} \tag{A19}$$

$$= \sum_{pq\in\bar{S}}\mathrm{Tr}\{\mathrm{Tr}_{\bar{w}}[(|p\rangle\langle q|_{\bar{S}} \otimes \mathbb{1}_{w,S})\sigma_L]\mathrm{Tr}_{\bar{w}}[(|q\rangle\langle p|_{\bar{S}} \otimes \mathbb{1}_{w,S})\sigma_L]\} \tag{A20}$$

$$= \mathrm{Tr}\{\mathrm{Tr}_S[\sigma_L]^2\} \tag{A21}$$

$$= \mathrm{Tr}[\tilde{\sigma}_{\bar{S},w}^2], \tag{A22}$$

where the last line comes from Lemma 1.

Similarly, for the second term in Eq. (A13), we have

$$\sum_{\substack{pq\\p'q'}} (\delta_{pq'})_{\bar{S}}(\delta_{qp'})_{\bar{S}}(\delta_{pq})_{S}(\delta_{p'q'})_{S}\mathrm{Tr}[\Psi_{pq}]\mathrm{Tr}[\Psi_{p'q'}]$$

$$= \mathrm{Tr}[\mathrm{Tr}_w[\tilde{\sigma}_{\bar{S},w}]^2]. \tag{A23}$$

Therefore, combining the results from Eqs. (A22) and (A23), we find

$$\sum_{\substack{pq\\p'q'}} (\delta_{pq'})_{\bar{S}}(\delta_{qp'})_{\bar{S}}(\delta_{pq})_{S}(\delta_{p'q'})_{S}\Delta\Psi_{pq}^{p'q'}$$

$$= \mathrm{Tr}\left[\mathrm{Tr}_w[\tilde{\sigma}_{\bar{S},w}^2] - \frac{\mathrm{Tr}_w[\tilde{\sigma}_{\bar{S},w}]^2}{d}\right] \tag{A24}$$

$$= D_{\mathrm{HS}}\left(\tilde{\sigma}_{\bar{S},w}, \tilde{\sigma}_{\bar{S}} \otimes \frac{\mathbb{1}}{d}\right), \tag{A25}$$

where in the last equation, we invoke Lemma 2. ∎

**Lemma 2.** [Monotonicity of the trace distance] Let $\mathcal{H}$ be a tensor product Hilbert space $\mathcal{H} = \mathcal{H}_A \otimes \mathcal{H}_B$, and let $R_{AB}$ and $S_{AB}$ be two operators on $\mathcal{H}$. Then, the following inequality holds:

$$D_T(R_{AB}, S_{AB}) \geq D_T(R_B, S_B), \tag{A26}$$

where $R_B = \mathrm{Tr}_A(R_{AB})$ and $S_B = \mathrm{Tr}_A(S_{AB})$. Here, $D_T(R_B, S_B) = \frac{1}{2}\|R_B - S_B\|_1$ is the trace distance, with $\|\cdot\|_1$ the Schatten 1- norm.

**Remark.** The monotonicity of the trace distance is usually proven when $R_{AB}$ and $S_{AB}$ are density matrices. The proof we present here is valid for any two matrices.

*Proof.*—By duality of Schatten norms [67], we have

$$\|R_B - S_B\|_1 = \max_{\Lambda_B \text{ s.t.} \|\Lambda_B\|_\infty = 1} \mathrm{Tr}[\Lambda_B(R_B - S_B)]. \tag{A27}$$

Therefore, there exists a matrix $\Lambda_B^*$ with $\|\Lambda_B\|_\infty = 1$ such that the maximum in Eq. (A27) is reached, i.e.,

$$\|R_B - S_B\|_1 = \mathrm{Tr}[\Lambda_B^*(R_B - S_B)]$$
$$= \mathrm{Tr}[(\mathbb{1}_A \otimes \Lambda_B^*)(R_{AB} - S_{AB})]$$
$$\leq \max_{\Lambda_{AB} \text{ s.t.} \|\Lambda_{AB}\|_\infty = 1} \mathrm{Tr}[\Lambda_{AB}(R_{AB} - S_{AB})]$$
$$= \|R_{AB} - S_{AB}\|_1. \tag{A28}$$

The inequality follows from Eq. (A27) and the fact that $\|\mathbb{1}_A \otimes \Lambda_B^*\|_\infty = 1$, while the last equality is simply another application of the Schatten norm duality.

We remark that an alternative proof, for the case where $R_{AB}$ and $S_{AB}$ are Hermitian, proceeds as follows. Let us begin by noting that one can write $R_{AB} - S_{AB} = P_{AB} - Q_{AB}$ where $P_{AB} \geq 0$ and $Q_{AB} \geq 0$, and $P_{AB}$ is orthogonal

to $Q_{AB}$. The orthogonality condition gives $|P_{AB} - Q_{AB}| = P_{AB} + Q_{AB}$. One then has

$$D_T(R_{AB}, S_{AB}) = \frac{1}{2}\mathrm{Tr}|P_{AB} - Q_{AB}| \tag{A29}$$

$$= \frac{1}{2}(\mathrm{Tr}P_{AB} + \mathrm{Tr}Q_{AB}) \tag{A30}$$

$$= \frac{1}{2}(\mathrm{Tr}P_B + \mathrm{Tr}Q_B) \tag{A31}$$

$$= \frac{1}{2}(\|P_B\|_1 + \|Q_B\|_1) \tag{A32}$$

$$\geq \frac{1}{2}(\|P_B - Q_B\|_1) \tag{A33}$$

$$= \frac{1}{2}(\|R_B - S_B\|_1), \tag{A34}$$

where $P_B = \mathrm{Tr}_A(P_{AB})$ and $Q_B = \mathrm{Tr}_A(Q_{AB})$. Note that the inequality in this proof follows from the triangle inequality for matrix norms. ∎

**Lemma 5.** Let $\mathcal{H}$ be a tensor product Hilbert space $\mathcal{H} = \mathcal{H}_A \otimes \mathcal{H}_B$, with $\dim H_A = d_A$ and $\dim H_B = d_B$, and let $R_{AB}$ be an operator on $\mathcal{H}$. Then, the following inequality holds:

$$D_{\mathrm{HS}}\left(R_{AB}, R_A \otimes \frac{\mathbb{1}_B}{d_B}\right) \geq \frac{D_{\mathrm{HS}}(R_B, \frac{\mathbb{1}}{d_B})}{d_B d_A}. \tag{A35}$$

*Proof.*—The equivalence of matrix norms gives us the following inequality between the Frobenius norm $\|M\|_2$ and the trace norm $\|M\|_1$ of a matrix $M$ with dimension $d$ [68]:

$$\|M\|_2 \leq \|M\|_1 \leq \sqrt{d}\|M\|_2. \tag{A36}$$

The first inequality comes from the monotonicity of the Schatten norms [67,69] and the second one from the Cauchy-Schwartz inequality. Since $D_{\mathrm{HS}}(R, S) = \|R - S\|_2^2$ and $D_T(R, S) = \frac{1}{2}\|R - S\|_1$, it follows from Eq. (A36) that

$$\frac{1}{4}D_{\mathrm{HS}}(R, S) \leq D_T(R, S)^2 \leq \frac{d}{4}D_{\mathrm{HS}}(R, S). \tag{A37}$$

Therefore, in our case,

$$D_{\mathrm{HS}}\left(R_{AB}, R_A \otimes \frac{\mathbb{1}_B}{d_B}\right) \geq \frac{4D_T(R_{AB}, R_A \otimes \frac{\mathbb{1}_B}{d_B})^2}{d_B d_A} \tag{A38}$$

$$\geq \frac{4D_T(R_B, \frac{1}{d_B})^2}{d_B d_A}$$

$$\geq \frac{D_{\mathrm{HS}}(R_B, \frac{1}{d_B})}{d_B d_A}, \tag{A39}$$

where the first and last inequalities come from the equivalence of matrix norms given by Eq. (A37), while the second inequality follows from the monotonicity of the trace distance as stated in Lemma 4. ∎

## APPENDIX B: PROOF OF THEOREM 1

In this Appendix, we present a proof of Theorem 1, which establishes a lower bound for the variance of the cost function partial derivative in terms of the coefficients of a graph obtained through the GRIM. Moreover, we also present a proof for Proposition 1. Let us restate the theorem here for convenience.

**Theorem 1.** Consider the QCNN cost function of Eq. (4) and let $W$ be a unitary in the first sublayer of the $\ell$th layer of $V(\boldsymbol{\theta})$. Moreover, let us assume that the unitaries $W_{ij}$, $W_A$, and $W_B$ in the convolutional and fully connected layers form independent 2-designs. Then, a lower bound of the variance of the cost function partial derivative, with respect to a parameter $\theta_\mu$ in unitary $W$, can be computed using the GRIM from the graph $\mathcal{G}_w$ as

$$\text{Var}[\partial_\mu C] \geq F_n(L, \ell), \qquad \text{(B1)}$$

with

$$F_n(L, \ell) = \frac{1}{9} \frac{\text{Tr}[H_\mu^2]\varepsilon_O\varepsilon_{\sigma_w}}{50^{L-\ell+1}} \sum_{\boldsymbol{g} \in P_\ell(\mathcal{G}_w)} \Lambda_{\boldsymbol{g}}. \qquad \text{(B2)}$$

Here, $\sigma_w = \text{Tr}_{\bar{w}}[\sigma]$ is the reduced operator $\sigma$ in the subsystem of the qubits that $W$ acts on. Moreover, $\varepsilon_O$, $P_\ell(\mathcal{G}_w)$, $\boldsymbol{g}$, and $\Lambda_{\boldsymbol{g}}$ are defined in Proposition 1.

### 1. Variance of the cost function partial derivative

Let us assume here that the block $W$ is in the $\ell$th layer of the QCNN. As we discuss in the main text, we can decompose $V$ as

$$V = V_R W V_L,$$

where $V_R$ contains all the gates in the forward light cone $\mathcal{L}_F$ of $W$, and where $V_L$ contains all remaining gates. Moreover, $W$ can also be decomposed as

$$W = W_A W_B,$$

where

$$W_A = \prod_{\eta \leq \mu} e^{-i\theta_\eta H_\eta} W_\eta, \qquad W_B = \prod_{\eta > \mu} e^{-i\theta_\eta H_\eta} W_\eta.$$

As shown in Ref. [36], the partial derivative of $C(\boldsymbol{\theta})$ with respect to $\theta_\mu$ can be written as

$$\partial_\mu C = \text{Tr}[W_A V_L \sigma V_L^\dagger W_A^\dagger [H_\mu, W_B^\dagger V_R^\dagger \tilde{O} V_R W_B]],$$

and if either $W_A$ or $W_B$ form a 1-design, we have

$$\langle \partial_\mu C \rangle = 0$$

and

$$\text{Var}[\partial_\mu C] = \frac{2\text{Tr}[H_\mu^2]}{225} \sum_{\substack{pq \\ p'q'}} \langle \Delta\Omega_{qp}^{q'p'} \rangle_{V_R} \langle \Delta\Psi_{pq}^{p'q'} \rangle_{V_L}. \qquad \text{(B3)}$$

Here the summation runs over all bit strings $p$, $q$, $p'$, $q'$ of length $2^{n-2}$. In addition, we define

$$\Delta\Omega_{qp}^{q'p'} = \text{Tr}[\Omega_{qp}\Omega_{q'p'}] - \frac{\text{Tr}[\Omega_{qp}]\text{Tr}[\Omega_{q'p'}]}{4},$$

$$\Delta\Psi_{pq}^{p'q'} = \text{Tr}[\Psi_{pq}\Psi_{p'q'}] - \frac{\text{Tr}[\Psi_{pq}]\text{Tr}[\Psi_{p'q'}]}{4},$$

where $\Omega_{qp}$ and $\Psi_{qp}$ are operators on $\mathcal{H}_w$ defined as

$$\Omega_{qp} = \text{Tr}_{\bar{w}}[(|\boldsymbol{p}\rangle\langle\boldsymbol{q}| \otimes \mathbb{1}_w)V_R^\dagger \tilde{O} V_R],$$

$$\Psi_{pq} = \text{Tr}_{\bar{w}}[(|\boldsymbol{q}\rangle\langle\boldsymbol{p}| \otimes \mathbb{1}_w)V_L \sigma V_L^\dagger].$$

Here, $\text{Tr}_{\bar{w}}$ indicates the trace over all qubits not in $\mathcal{H}_w$, with $\mathcal{H}_w$ being the Hilbert space associated with the qubits that $W$ acts on.

In what follows, we employ the GRIM to compute the contributions of the expectation values $\langle \Delta\Omega_{qp}^{q'p'} \rangle_{V_R}$ and $\langle \Delta\Psi_{pq}^{p'q'} \rangle_{V_L}$ to the variance of the cost function partial derivative.

### 2. Integration over the unitaries in $V_R$ via the GRIM

#### a. Grouping the unitaries in modules

Let us now focus on the term $\langle \Delta\Omega_{qp}^{q'p'} \rangle_{V_R}$. As we discuss in the main text, here we employ the GRIM, whose first step is to group the unitaries in $\mathcal{L}_F$ into modules $M$. Moreover, as we show in Fig. 8, for the QCNN architecture one needs only three basic modules: a center module $M_{\mathcal{C}}$, a middle module $M_{\mathcal{M}}$, and an edge module $M_{\mathcal{E}}$. Note that different choices of $W$ lead to different modules being employed. Surprisingly, for any $W$ we have one of the following three cases:

   (i) Case 1: The light cone $\mathcal{L}_F$ can be covered by $\ell$ modules $M_{\mathcal{C}}$.
  (ii) Case 2: The light cone $\mathcal{L}_F$ can be covered by $(\ell - 1)$ modules $M_{\mathcal{E}}$ and a module $M_{\mathcal{C}}$.
 (iii) Case 3: The light cone $\mathcal{L}_F$ can be covered by $\ell'$ modules $M_{\mathcal{M}}$, $\ell''$ modules $M_{\mathcal{E}}$, and a module $M_{\mathcal{C}}$ such that $\ell' + \ell'' + 1 = \ell$.

Let us make the following important remarks. First, note that the *final* module is always $M_{\mathcal{C}}$. Second, let us recall that $W$ can be in the first or in the second sublayer. As indicated in Fig. 8(a), if $W$ is in the first sublayer, then $W$ does not belong to any module. However, if $W$ is in the second sublayer, then it is always one of the unitaries in a given module [see Fig. 8(b)]. This can be alternatively thought as introducing additional basic *initial* modules corresponding to taking $M_{\mathcal{C}}$, $M_{\mathcal{M}}$, and $M_{\mathcal{E}}$ and removing a unitary from the second sublayer.
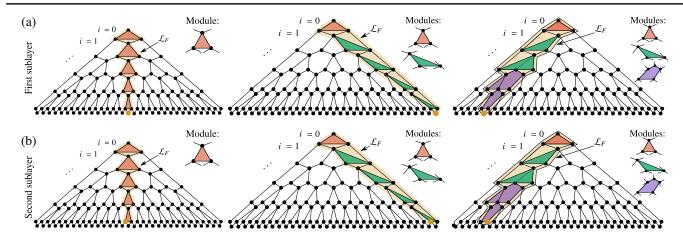
FIG. 8.   Examples of the first step of the GRIM. Here we can cover all of the forward light cone $\mathcal{L}_F$ using three basic modules repeated $\ell$ times. Panels (a) and (b), respectively, correspond to the cases when the unitary $W$ is in first sublayer and second sublayer of the $\ell$th QCNN layer. In all cases, $W$ is indicated by a larger orange circle, and the forward light cone is shaded in yellow. For the panels in (a), $W$ is not included in any module, while in (b) $W$ belongs to the first module $M$.

More formally, the first step of the GRIM implies that $V_R$ can always be decomposed into a set of $\ell$ modules:

$$V_R = V_R^{(\ell)} V_R^{(\ell-1)} \cdots V_R^{(1)}, \tag{B4}$$

where $V_R^{(i)} \in \{M_\mathcal{C}, M_\mathcal{E}, M_\mathcal{M}\}$ for $2 \leq i \leq \ell - 1$, while $V_R^{(1)}$ and $V_R^{(\ell)} = M_\mathcal{C}$ are initial and final modules. To integrate the initial module $V_R^{(1)}$, it is convenient to rewrite $\Omega_{qp}$ as

$$\Omega_{qp} = \mathrm{Tr}_{\bar{w}}[(|\boldsymbol{p}\rangle\langle\boldsymbol{q}| \otimes \mathbb{1}_w)(V_R^{(1)})^\dagger \cdots (V_R^{(\ell-1)})^\dagger (V_R^{(\ell)})^\dagger O V_R^{(\ell)} V_R^{(\ell-1)} \cdots V_R^{(1)}] \tag{B5}$$

$$= \mathrm{Tr}_{\bar{w}}[(|\boldsymbol{p}\rangle\langle\boldsymbol{q}| \otimes \mathbb{1}_w)(V_R^{(1)})^\dagger \tilde{O}_{\ell-1} V_R^{(1)}], \tag{B6}$$

where we define

$$\tilde{O}_i = (V_R^{(\ell-i+1)})^\dagger \cdots (V_R^{(\ell-1)})^\dagger (V_R^{(\ell)})^\dagger O V_R^{(\ell)} V_R^{(\ell-1)} \cdots V_R^{(\ell-i+1)}. \tag{B7}$$

Finally, let $S_{\bar{w}}$ be the set of qubits in $\mathcal{H}_{\bar{w}}$; that is, the set of qubits that $W$ acts trivially on. We decompose $S_{\bar{w}}$ as the disjoint union

$$S_{\bar{w}} = \mathcal{L}_1 \cup, \dots, \mathcal{L}_\ell \cup \bar{\mathcal{L}}, \tag{B8}$$

where $\mathcal{L}_k$ represents all the qubits in $S_{\bar{w}}$ that $V_R^{(k)}$ acts nontrivially on. Moreover, we define

$\bar{\mathcal{L}} = S_{\bar{w}}\backslash(\mathcal{L}_1 \cup, \dots, \cup \mathcal{L}_\ell)$ as the set of all the qubits that are not in the forward light cone of $W$. Using this decomposition, we can rewrite the partial trace of Eq. (B5) as

$$\Omega_{qp} = \mathrm{Tr}_{\mathcal{L}_1, \dots \mathcal{L}_\ell, \bar{\mathcal{L}}}[(|\boldsymbol{p}\rangle\langle\boldsymbol{q}| \otimes \mathbb{1}_w)(V_R^{(1)})^\dagger \tilde{O}_{\ell-1} V_R^{(1)}]. \tag{B9}$$

In Fig. 9, we show the tensor network representation of $\Omega_{qp}$ and of $\Delta\Omega_{qp}^{q'p'}$ for the case when $V_R^{(1)} = M_\mathcal{M}$. In the next section, we show how to generate the graph $\mathcal{G}_w$ via the GRIM.
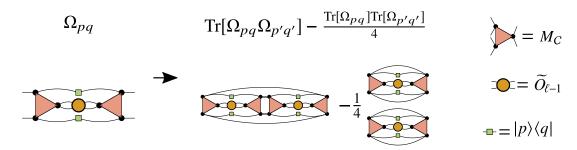


FIG. 9.   Tensor network representation of $\Omega_{qp}$ and $\Delta\Omega_{qp}^{q'p'}$. Here we consider the case where $W$ is in the middle of the $\ell$th layer.

### b. Basic notation

Here we introduce the basic notation employed to form the GRIM graph. For simplicity, let us consider the tensor network of $\Delta\Omega_{qp}^{q'p'}$ as shown in Fig. 9, and let us consider the task of integrating the blocks in $V_R^{(1)}$. Employing the Weingarten calculus of Eqs. (A2) and (A3), we can express the result of $\langle\Delta\Omega_{qp}^{q'p'}\rangle_{V_R^{(1)}}$ as a linear combination of all possible contractions of the $w$ and $\mathcal{L}_1$ wires. Noting that $\Delta\Omega_{qp}^{q'p'} = N_1(\tilde{O}_\ell)$, with $N_1(\tilde{O}) = T_\varnothing(\tilde{O}) - \frac{1}{4}T_w(\tilde{O})$ [see Eqs. (B11) and (B12)], we write

$$\int_{V_R^{(1)}} \Delta\Omega_{qp}^{q'p'} d\mu = \int_{V_R^{(1)}} N_1(\tilde{O}_\ell) d\mu$$

$$= \sum_\beta \sum_{s'\in\mathcal{P}(\mathcal{L}_1)} a_{1,\beta,s'} p_{1,\beta,s'} N_\beta(\tilde{O}_{\ell-1}), \quad (B10)$$

where we denote $\int_{V_R^{(1)}} d\mu = \int, ..., \int dW_1, ..., dW_m$ as the integral with respect to the Haar measure over the unitary group for all gates $\{W_{\text{init}}\}_{i=1}^m$ appearing in $V_R^{(1)}$. The coefficients $a_{1,\beta,s'}$ are real numbers to be found, the nonzero of which characterize the module that we are integrating. Moreover, we remark that $\mathcal{P}(\mathcal{L}_1)$ and $\mathcal{P}(\mathcal{L}_1 \cup w)$, respectively, denote the power sets of the qubits in $\mathcal{L}_1$ and $\mathcal{L}_1 \cup w$. Hence, the summations in Eqs. (B10) and (B11) are taken over all the subsets $s'$ and $s$ of $\mathcal{L}_1$ and $\mathcal{L}_1 \cup w$, respectively. In addition, here we define the *nodes*

$$N_\beta(\tilde{O}_{\ell-1}) = \sum_{s\in\mathcal{P}(\mathcal{L}_1\cup w)} c_{\beta,s} T_s(\tilde{O}_{\ell-1}), \quad (B11)$$

where $c_{\beta,s}$ are real coefficients that characterize the node, and we define the contraction of the operator $\tilde{O}_{\ell-1}$,

$$T_s(\tilde{O}_{\ell-1}) = \text{Tr}\{\text{Tr}_{s,\mathcal{L}_2,...,\mathcal{L}_\ell,\bar{\mathcal{L}}}[(|\boldsymbol{p}\rangle\langle\boldsymbol{q}| \otimes \mathbb{1}_w)\tilde{O}_{\ell-1}]\text{Tr}_{s,\mathcal{L}_2,...,\mathcal{L}_\ell,\bar{\mathcal{L}}}[(|\boldsymbol{p'}\rangle\langle\boldsymbol{q'}| \otimes \mathbb{1}_w)\tilde{O}_{\ell-1}]\}. \quad (B12)$$

Here, $|\boldsymbol{p}\rangle\langle\boldsymbol{q}|$ and $|\boldsymbol{p'}\rangle\langle\boldsymbol{q'}|$ are operators over the qubits in $S_{\bar{w}}\backslash\mathcal{L}_1$. The contribution of the projectors $|\boldsymbol{p}\rangle\langle\boldsymbol{q}|$ and $|\boldsymbol{p'}\rangle\langle\boldsymbol{q'}|$ over the qubits in $\mathcal{L}_1$ is given by the operators $p_{1,\beta,s}$, which are defined as

$$p_{1,\beta,s} = \text{Tr}\{\text{Tr}_s[|\boldsymbol{p}\rangle\langle\boldsymbol{q}|_{\mathcal{L}_1}]\text{Tr}_s[|\boldsymbol{p'}\rangle\langle\boldsymbol{q'}|_{\mathcal{L}_1}]\} \quad (B13)$$

$$= \delta_{(\boldsymbol{pq})_s}\delta_{(\boldsymbol{p'q'})_s}\delta_{(\boldsymbol{pq})_{\bar{s}}}\delta_{(\boldsymbol{p'q})_{\bar{s}}}, \quad (B14)$$

where $\bar{s} = \mathcal{L}_1\backslash s$. We refer the reader to Fig. 10(a) for a graphical representation of the operator $p_{1,\beta,s}$.

Here we remark that $a_{1,\beta,s'}$ and $c_{\beta,s}$ are not uniquely defined for a given integral, since for every $s'$, we can replace $c_{\beta,s}$ by $\gamma_\beta c_{\beta,s}$ and $a_{1,\beta,s'}$ by $a_{1,\beta,s'}/\gamma_\beta$, where $\gamma_\beta$ can be any nonzero number:

$$\sum_\beta \sum_{s'\in\mathcal{P}(\mathcal{L}_1)} a_{1,\beta,s'} p_{1,\beta,s'} \sum_{s\in\mathcal{P}(\mathcal{L}_1\cup w)} c_{\beta,s} T_s(\tilde{O}_{\ell-1})$$

$$= \sum_\beta \sum_{s'\in\mathcal{P}(\mathcal{L}_1)} \frac{a_{1,\beta,s'}}{\gamma_\beta} p_{1,\beta,s'} \sum_{s\in\mathcal{P}(\mathcal{L}_1\cup w)} \gamma_\beta c_{\beta,s} T_s(\tilde{O}_{\ell-1}).$$

$$(B15)$$

To fix this freedom, we henceforth impose that $c_{\beta,\varnothing} = 1$ or $-1$ and that $a_{1,\beta,s'} \geq 0$. The choice of $a_{1,\beta,s'} \geq 0$ guarantees that the edge coefficients of the GRIM graph are always positive.

### c. Construction of the graph

Employing the notation introduced in the previous section, we can now construct the structure of the graph $\mathcal{G}_w$, of which we provide a canonical version in Fig. 10(b). The recursive procedure is defined as follows:

(a)

$$p_{\alpha,\beta,s} = s\,\overline{\underline{\phantom{x}}\,\square\,\square\,\underline{\phantom{x}}}\,s$$

(b)
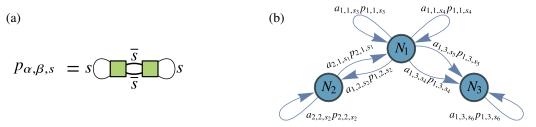


FIG. 10. (a) Schematic representation of the $p_{\alpha,\beta,s}$ coefficients. The contractions of the operators $|\boldsymbol{p}\rangle\langle\boldsymbol{q}|$ lead to the terms $p_{\alpha,\beta,s}$ according to Eq. (B13). (b) Representative graph obtained after integrating the modules in the forward light cone $\mathcal{L}_F$. The node $N_\alpha$ is connected to the node $N_\beta$ via an (oriented) arrow. Each edge is associated with an operator $e_{\alpha,\beta,s} = a_{\alpha,\beta,s} p_{\alpha,\beta,s}$.

(i) We define an initial node $N_1$ defined as

$$N_1(\tilde{O}) = T_\varnothing(\tilde{O}) - \frac{1}{4}T_w(\tilde{O}), \qquad (B16)$$

for any operator $\tilde{O}$. Note that $N_1$ is simply a particular case of Eq. (B11). As we describe in the previous section, we integrate the unitaries in $V_R^{(1)}$ and obtain a result of the form (B10). In the graph, we represent this by drawing an oriented edge (an arrow) from the initial node $N_1$ to the nodes $N_\beta$ that appear on the right-hand side of Eq. (B10).

(ii) Let $N_\alpha$ be a node in the graph obtained by integrating the $(i-1)$th modules. To construct the next nodes in the graph, we integrate $N_\alpha$ over the module $V_R^{(i)}$. Employing Eq. (B10), we can write the result as

$$\int_{V_R^{(i)}} N_\alpha(\tilde{O}^{(i)})d\mu$$
$$= \sum_\beta \sum_{s' \in \mathcal{P}(\mathcal{L}_i)} a_{\alpha,\beta,s'} p_{\alpha,\beta,s'} N_\beta(\tilde{O}^{(i+1)})$$
$$= \sum_\beta \sum_{s' \in \mathcal{P}(\mathcal{L}_i)} e_{\alpha,\beta,s'} N_\beta(\tilde{O}^{(i+1)}). \qquad (B17)$$

As shown in Fig. 10(b), for any nonzero $a_{\alpha,\beta,s'} p_{\alpha,\beta,s'}$ we associate a new edge (arrow) from $N_\alpha$ with the nodes on the right-hand-side of Eq. (B17). Here we define for convenience the operator

$$e_{\alpha,\beta,s'} = a_{\alpha,\beta,s'} p_{\alpha,\beta,s'}. \qquad (B18)$$

Applying the aforementioned process $\ell$ times, we can obtain the basic structure of the graph $\mathcal{G}_w$, which is composed of a set of nodes $\{N_\alpha\}$ and associated oriented edges. Let us make two important remarks. First, each edge of the graph is now associated with an operator $e_{\alpha,\beta,s'}$. Below, we show how to obtain the real edge coefficients $\lambda_{\alpha,\beta}$ from the $e_{\alpha,\beta,s'}$. Moreover, we also define an equivalence relation between nodes such that $N_{\alpha_1}$ and $N_{\alpha_2}$ correspond to the same node if they apply the same wire contractions, independent of the qubit that the wire represents.

Next, let us define a walk $\hat{g}$ through the graph as a sequence of connected edges

$$\hat{g} = (e_{\alpha_1,\alpha_2,s_1'}, e_{\alpha_2,\alpha_3,s_2'}, \ldots), \qquad (B19)$$

such that

(i) the operator $e_{\alpha_i,\alpha_{i+1},s_i}$ is associated with one of the edges connecting $N_{\alpha_i}$ to $N_{\alpha_{i+1}}$;
(ii) the walk starts at the node $N_1$ of Eq, (B16), i.e., $\alpha_1 = 1$;
(iii) as we discuss below, in all QCNN graphs the walk ends at a node $N_1$ evaluated at the operator $O$.

We denote the set of all such paths of length $\ell$ as $\hat{P}_\ell$.

From the ensuing graph and the previous results we can now compute the expectation value $\langle \Delta\Omega_{qp}^{q'p'} \rangle_{V_R}$. Consider a walk $\hat{g} \in \hat{P}_\ell$. Then, let us define the operator

$$Q(\hat{g}) = \prod_{e_{\alpha,\beta,s} \in \hat{g}} e_{\alpha,\beta,s}, \qquad (B20)$$

that is, the product of the weights along the walk. We can then write

$$\langle \Delta\Omega_{qp}^{q'p'} \rangle_{V_R} = \int_{V_R^{(\ell)}} \cdots \int_{V_R^{(1)}} \Delta\Omega_{qp}^{q'p'} d\mu_1 \cdots d\mu_\ell$$
$$= N_1(O) \sum_{\hat{g} \in \hat{P}_\ell} Q(\hat{g}). \qquad (B21)$$

This equation can simplified by noting that, by definition,

$$N_1(O) = T_\varnothing(O) - \frac{1}{4}T_w(O)$$
$$= \mathrm{Tr}\{\mathrm{Tr}_{\bar{L}}[(|\boldsymbol{p}\rangle\langle\boldsymbol{q}| \otimes \mathbb{1}_w)O]\mathrm{Tr}_{\bar{L}}[(|\boldsymbol{p}'\rangle\langle\boldsymbol{q}'| \otimes \mathbb{1}_w)O]\}$$
$$\quad - \frac{1}{4}\mathrm{Tr}[(|\boldsymbol{p}\rangle\langle\boldsymbol{q}| \otimes \mathbb{1}_w)O]\mathrm{Tr}[(|\boldsymbol{p}'\rangle\langle\boldsymbol{q}'| \otimes \mathbb{1}_w)O]$$
$$= \mathrm{Tr}[|\boldsymbol{p}\rangle\langle\boldsymbol{q}|]\mathrm{Tr}[|\boldsymbol{p}'\rangle\langle\boldsymbol{q}'|]\left(\mathrm{Tr}[O^2] - \frac{1}{4}\mathrm{Tr}[O]^2\right)$$
$$= (\delta_{\boldsymbol{pq}})_{\bar{L}}(\delta_{\boldsymbol{p}'\boldsymbol{q}'})_{\bar{L}}\varepsilon_O, \qquad (B22)$$

where $\varepsilon_O = \mathrm{Tr}[O^2] - \frac{1}{4}\mathrm{Tr}[O]^2$ is defined in the main text. Therefore, we have

$$\langle \Delta\Omega_{qp}^{q'p'} \rangle_{V_R} = (\delta_{\boldsymbol{pq}})_{\bar{L}}(\delta_{\boldsymbol{p}'\boldsymbol{q}'})_{\bar{L}}\varepsilon_O \sum_{\hat{g} \in \hat{P}_\ell} Q(\hat{g}). \qquad (B23)$$

The next step in the GRIM is to simplify the graph by lower bounding Eq. (B23) and associating with each vertex a real positive coefficient $\lambda_{\alpha\beta}$ in place of each operator $e_{\alpha,\beta,s}$ that appears in $Q(\hat{g})$. We remark that below we analyze the three possible cases for the placement of $W$.

### d. Simplification of the graph

Let us first recall that all the $p_{\alpha,\beta,s}$ obtained from integrating the $k$th module can be written in terms of the operators $(\delta_{\boldsymbol{pq}'})_{\bar{s}_k}(\delta_{\boldsymbol{qp}'})_{\bar{s}_k}(\delta_{\boldsymbol{pq}})_{s_k}(\delta_{\boldsymbol{p}'\boldsymbol{q}'})_{s_k}$ where $\bar{s}_k \cup s_k = \mathcal{L}_k$ and $\bar{s}_k \cap s_k = \varnothing$. Hence, from Eq. (B23) we can write

$$(\delta_{\boldsymbol{pq}})_{\bar{L}}(\delta_{\boldsymbol{p}'\boldsymbol{q}'})_{\bar{L}} \prod_{e_{\alpha,\beta,s} \in \hat{g}} p_{\alpha,\beta,s} = (\delta_{\boldsymbol{pq}'})_{\bar{S}_P}(\delta_{\boldsymbol{qp}'})_{\bar{S}_P}(\delta_{\boldsymbol{pq}})_{S_P}(\delta_{\boldsymbol{p}'\boldsymbol{q}'})_{S_P},$$
$$(B24)$$

where $S_P$ is the disjoint union of all the $s_k$ and $\bar{\mathcal{L}}$, and where $\bar{S}_P = \bar{w}\backslash S_P$.

Combining Eqs. (B3), (B23), and (B24), we find

$$\sum_{\substack{pq \\ p'q'}} \langle \Delta\Omega_{qp}^{q'p'} \rangle_{V_R} \langle \Delta\Psi_{pq}^{p'q'} \rangle_{V_L} = \varepsilon_O \sum_{\hat{g}\in\hat{P}_\ell} \left( \prod_{e_{\alpha,\beta,s}\in\hat{g}} a_{\alpha,\beta,s} \right) \sum_{\substack{pq \\ p'q'}} pq'(\delta_{pq'})_{\bar{S}_P}(\delta_{qp'})_{\bar{S}_P}(\delta_{pq})_{S_P}(\delta_{p'q'})_{S_P}\langle\Delta\Psi_{pq}^{p'q'}\rangle_{V_L} \tag{B25}$$

$$= \varepsilon_O \sum_{\hat{g}\in\hat{P}_\ell} \left( \prod_{e_{\alpha,\beta,s}\in\hat{g}} a_{\alpha,\beta,s} \right) \left\langle D_{\mathrm{HS}}\left( \tilde{\sigma}_{\bar{S}_P,w}, \tilde{\sigma}_{\bar{S}_P} \otimes \frac{\mathbb{1}}{4} \right) \right\rangle_{V_L} \tag{B26}$$

$$\geq \varepsilon_O \sum_{\hat{g}\in\hat{P}_\ell} \left( \prod_{e_{\alpha,\beta,s}\in\hat{g}} a_{\alpha,\beta,s} \right) \frac{1}{4} \left\langle D_{\mathrm{HS}}\left( \tilde{\sigma}_w, \frac{\mathbb{1}}{4}\mathrm{Tr}[\tilde{\sigma}] \right) \right\rangle_{V_L} \frac{1}{2^{|\bar{S}_P|}} \tag{B27}$$

$$= \frac{\varepsilon_O}{4} \langle\varepsilon_{\tilde{\sigma}_w}\rangle_{V_L} \sum_{\hat{g}\in\hat{P}_\ell} \left( \prod_{e_{\alpha,\beta,s}\in\hat{g}} a_{\alpha,\beta,s} \right) \frac{1}{2^{\sum_{e_{\alpha,\beta,s}\in\hat{g}}|\bar{s}|}} \tag{B28}$$

$$= \frac{\varepsilon_O}{4} \langle\varepsilon_{\tilde{\sigma}_w}\rangle_{V_L} \sum_{\hat{g}\in\hat{P}_\ell} \left( \prod_{e_{\alpha,\beta,s}\in\hat{g}} \frac{a_{\alpha,\beta,s}}{2^{|\bar{s}|}} \right), \tag{B29}$$

where we denote $|s|$ as the number of qubits in the set $s$. The first equality arises from the fact that $a_{\alpha,\beta,s}$ are independent of $p, q, p', q'$. The second equality follows from Lemma 3. Then, the inequality is obtained by invoking Lemma 5 (equivalence of the norms and the monotonicity of the partial trace). In Eq. (B28), we use the definition of $|\bar{S}_P|$. Finally, in the last line we simply group terms into a single product.

We see from Eq. (B29) that every coefficient of the form $(\delta_{pq'})_{\bar{s}}(\delta_{pq'})_{\bar{s}}$ induces a factor $\frac{1}{2^{|s|}}$ in the edge coefficients. That leads us to a new simplified version of our graph, where each edge operator $e_{\alpha,\beta,s}$ is replaced by an edge coefficient $\lambda_{\alpha\beta}$:

$$e_{\alpha,\beta,s} = a_{s_{\alpha,\beta,s}} p_{\alpha,\beta,s} \rightarrow \lambda_{\alpha\beta} = \sum_s \frac{a_{\alpha,\beta,s}}{2^{|\bar{s}|}}. \tag{B30}$$

Moreover, two edges connecting two given nodes $N_\alpha$ and $N_\beta$, with associated coefficients $e_{\alpha,\beta,s}$ and $e_{\alpha,\beta,s'}$, can be factored into a single edge in Eq. (B29). The associated coefficient is the sum of the $\lambda_{\alpha\beta}$ corresponding to the two edges.

We denote $\mathcal{G}_w$ as the final graph consisting of the simplified $\lambda_{\alpha\beta}$ coefficients, and we denote $P_\ell(\mathcal{G}_w)$ as the set of paths in this graph. Hence, the final lower bound can be written,

$$\sum_{\substack{pq \\ p'q'}} \langle\Delta\Omega_{qp}^{q'p'}\rangle_{V_R} \langle\Delta\Psi_{pq}^{p'q'}\rangle_{V_L} \geq \frac{\varepsilon_O}{4} \langle\varepsilon_{\tilde{\sigma}_w}\rangle_{V_L} \sum_{g\in P_\ell(\mathcal{G}_w)} \Lambda_g, \tag{B31}$$

where we define the sequence of connected edges

$$g = (\lambda_{\alpha_1,\alpha_2}, \lambda_{\alpha_2,\alpha_3}, \ldots) \tag{B32}$$

and

$$\Lambda_g = \prod_{\lambda_{\alpha\beta}\in g} \lambda_{\alpha\beta}.$$

Note that this constitutes a proof for Proposition 1.

### 3. Integration over the unitaries in $V_L$ via the GRIM

#### a. Method

In this section, we introduce the general method employed to compute the expectation value $\langle\varepsilon_{\tilde{\sigma}_w}\rangle_{V_L}$. In contrast to the forward light cone, as shown in Fig. 11(a), the width of the backward light cone $\mathcal{L}_B$ is not bounded and instead grows with the number of layers. The goal here is to employ the GRIM to reduce the number of gates that need to be integrated to compute $\langle\varepsilon_{\tilde{\sigma}_w}\rangle_{V_L}$.

First, let us remark that since we are interested in the operator $\tilde{\sigma}_w = \mathrm{Tr}_{\bar{w}}[V_L\sigma V_L^\dagger]$, all the unitaries in $V_L$ which are not in $\mathcal{L}_B$ will compile to identity [see Figs. 11(a) and 11(b)]. Defining $V_{\mathcal{L}_B}$ as the unitary containing the blocks in $\mathcal{L}_B$, we then have that $\langle\varepsilon_{\tilde{\sigma}_w}\rangle_{V_L} = \langle\varepsilon_{\tilde{\sigma}_w}\rangle_{V_{\mathcal{L}_B}}$.

As depicted in Fig. 11(a), the next step is to draw an effective backward light cone $\tilde{\mathcal{L}}_B$ and to define the unitary $V_{\tilde{\mathcal{L}}_B}$ which contains all the gates in $\tilde{\mathcal{L}}_B$. This effective light cone consists of the repetition of $M_C$ modules, starting after the integration of an initial unitary if we are in the second sublayer, as shown in Figs. 12 and 13. A method where we only have to keep track of the reduced light cone $\tilde{\mathcal{L}}_B$. It works by sequentially removing gates outside of $\tilde{\mathcal{L}}_B$ that compile to identity.

Here, we follow again the first step of GRIM to decompose $V_{\tilde{\mathcal{L}}_B}$ into a series of $(L-\ell)$ modules:

$$V_{\tilde{\mathcal{L}}_B} = V_{\tilde{\mathcal{L}}_B}^{(\ell+1)} \cdots V_{\tilde{\mathcal{L}}_B}^{(L-1)} V_{\tilde{\mathcal{L}}_B}^{(L)}. \tag{B33}$$
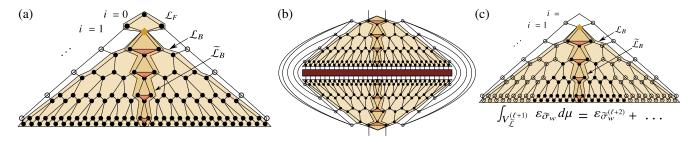
FIG. 11.    (a) Schematic representation of the backward light cone $\mathcal{L}_B$ (shaded in yellow) and the effective light cone $\tilde{\mathcal{L}}_B$ (shaded in a darker yellow). Here, the width of $\mathcal{L}_B$ increases with the number of layers. To avoid integrating all the unitaries in $\mathcal{L}_B$, we define an effective light cone $\tilde{\mathcal{L}}_B$ which can be covered by modules according to the GRIM. (b) Tensor network representation of the unitary $\tilde{\sigma}_w$. The large red rectangular tensor corresponds to $\sigma$. Here we can see that the unitaries in $V_L$ which are not in $\mathcal{L}_B$ simplify to identity when computing $\text{Tr}_{\bar{w}}[V_L \sigma V_L^\dagger]$. We mark such identities with an unshaded circle. (c) Integrating the unitaries in $V_{\tilde{\mathcal{L}}_B}^{(L-1)}$ leads to a summation of terms according to Eq. (B35). Here, one of these terms corresponds to $\varepsilon_{\tilde{\sigma}_w^{(\ell+2)}}$. As schematically shown, in the computation of $\varepsilon_{\tilde{\sigma}_w^{(\ell+2)}}$ there are many gates that simplify to identity. These gates are indicated here as shallow circles.
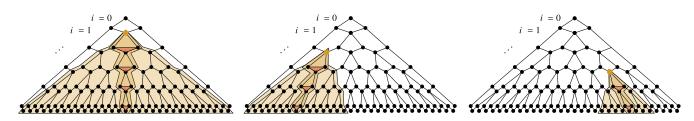


FIG. 12.    Effective backward light cone $\tilde{\mathcal{L}}_B$ for the case when $W$ is in the first sublayer or the $\ell$th layer of the QCNN. Here, $\tilde{\mathcal{L}}_B$ can be covered by $(L - \ell)$ modules $M_{\mathcal{M}}$.
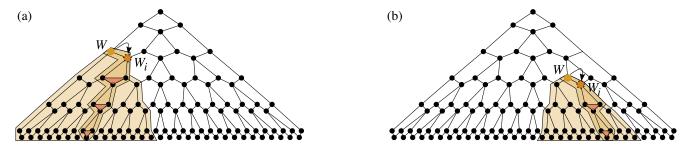


FIG. 13.    (a) The unitary $W$ is in the edge of the second sublayer. Here all unitaries in the backward light cone $\tilde{\mathcal{L}}_B$ can be grouped in modules $M_{\mathcal{M}}$ except for the initial unitary $W_{\text{init}}$. (b) The unitary $W$ is in the second sublayer but not in its edge. The backward light-cone structure is similar to the one in panel (a).

Equation (B33) also allows us to define the operators

$$\tilde{\sigma}^{(\ell+k)} = V_{\tilde{\mathcal{L}}_B}^{(\ell+k)} \cdots V_{\tilde{\mathcal{L}}_B}^{(L)} \sigma (V_{\tilde{\mathcal{L}}_B}^{(L)})^\dagger \cdots (V_{\tilde{\mathcal{L}}_B}^{(\ell+k)})^\dagger, \quad \text{(B34)}$$

and we write $\tilde{\sigma}_w^{(\ell+k)} = \text{Tr}_{\bar{w}}[\tilde{\sigma}^{(\ell+k)}]$.

As usual, the second step of the GRIM is to integrate the unitaries in $V_{\tilde{\mathcal{L}}_B}^{(\ell')}$, which again leads to a recursion of the form

$$\int_{V_{\tilde{\mathcal{L}}_B}^{(\ell+1)}} \tilde{N}_\alpha(\tilde{\sigma}_w^{(\ell')}) d\mu = \sum_\beta \sum_{s' \in \mathcal{P}(\mathcal{L}_i)} a_{\alpha,\beta,s'} \tilde{N}_\beta(\tilde{\sigma}_w^{(\ell'+1)}), \quad \text{(B35)}$$

where here the nodes $\tilde{N}_\alpha$ contain no $|p\rangle\langle q|$ operators due to the fact that $\varepsilon_{\tilde{\sigma}_w}$ is independent of $p, q, p'$, and $q'$. Moreover, as we explicitly show below for all three possible scenarios for placement of $W$, there will always be specific tensor contractions $N_\alpha$ where the unitaries not in $V_{\tilde{\mathcal{L}}_B}$ compile to identify. Such a procedure is schematically shown in Fig. 11(c) for the middle module, where computing $\varepsilon_{\sigma_w^{(\ell+2)}}$ leads to many more blocks in $V_{\mathcal{L}_B}$ being simplified

and compiling to the identity. Hence, by sequentially repeating this procedure, we can obtain a lower bound for $\langle \varepsilon_{\tilde{\sigma}_w} \rangle_{V_L}$ without needing to integrate all gates in $\mathcal{L}_B$, but just those in $\tilde{\mathcal{L}}_B$.

In the next section, we explicitly show this procedure for all possible choices of $W$ gate placement.

### b. W in the first sublayer

Here we consider the first case when $W$ is in the first sublayer of the $\ell$th layer. This case is schematically shown in Fig. 12, where we can see that each unitary in $\tilde{\mathcal{L}}_B$ can be covered by $(L - \ell)$ modules $M_{\mathcal{C}}$. By explicitly integrating the unitaries in the first module $M_{\mathcal{C}}$ of $\tilde{\mathcal{L}}_B$, we find

$$\int_{M_{\mathcal{M}}} \varepsilon_{\tilde{\sigma}_w} d\mu = \frac{1}{50} D_{\mathrm{HS}} \left( \tilde{\sigma}_w^{(\ell+2)}, \frac{\mathbb{1}}{4} \mathrm{Tr}[\sigma] \right)$$
$$+ \sum_{i,j} \tilde{c}_{ij} D_{\mathrm{HS}} \left[ \tilde{T}_i(\tilde{\sigma}^{(\ell+2)}), \tilde{T}_j(\tilde{\sigma}^{(\ell+2)}) \right], \quad \text{(B36)}$$

where $\tilde{c}_{i,j} \geq 0 \, \forall \, i, j$. Here, $\tilde{T}_i(\tilde{\sigma}_w^{(\ell+1)})$ are different contractions obtained from the operator $\tilde{\sigma}_w^{(\ell+1)}$, and $\tilde{c}_i$ are real positive coefficients. Since the Hilbert-Schmidt distance is always positive, we have that the lower bound always holds:

$$\int_{M_{\mathcal{M}}} \varepsilon_{\tilde{\sigma}_w} d\mu \geq \frac{1}{50} D_{\mathrm{HS}} \left( \tilde{\sigma}_w^{(\ell+2)}, \frac{\mathbb{1}}{4} \mathrm{Tr}[\sigma] \right) \quad \text{(B37)}$$

$$= \frac{1}{50} \varepsilon_{\tilde{\sigma}_w^{(\ell+2)}}. \quad \text{(B38)}$$

As we discuss in the previous section, repeating this procedure $(L - \ell)$ times and recursively applying the lower bound of Eq. (B36) leads to

$$\langle \varepsilon_{\tilde{\sigma}_w} \rangle_{V_L} \geq \left( \frac{1}{50} \right)^{L-\ell} D_{\mathrm{HS}} \left( \sigma_w, \frac{\mathbb{1}}{4} \mathrm{Tr}[\sigma] \right) \quad \text{(B39)}$$

$$= \left( \frac{1}{50} \right)^{L-\ell} \varepsilon_{\sigma_w}. \quad \text{(B40)}$$

Here we remark that combining Eqs. (B3), (B31), and (B40) leads to the proof of Theorem 1. In what follows, we consider the case when $W$ is in the second sublayer.

### c. W in the edge of the second sublayer

As shown in Fig. 13(a), here we consider the case when the unitary $W$ is either the first or the last unitary of the second sublayer of the $\ell$th layer of the QCNN. In order to group the unitaries in $\tilde{\mathcal{L}}_B$ into modules $M_{\mathcal{M}}$ as we did in the previous section, we first have to integrate the unitary that is adjacent to $W$ and in the first sublayer. This unitary is

indicated in Fig. 13(a) by a larger circle with a dashed edge. We call the unitary $W_{\mathrm{init}}$ as it is the first unitary that needs to be integrated in $\tilde{\mathcal{L}}_B$.

Explicitly, we find

$$\int_{W_{\mathrm{init}}} \varepsilon_{\tilde{\sigma}_w} d\mu = \frac{1}{5} D_{\mathrm{HS}} \left( \tilde{\sigma}_{w_{\mathrm{init}}}^{(\ell+2)}, \frac{\mathbb{1}}{4} \mathrm{Tr}[\sigma] \right)$$
$$+ \sum_{i,j} \tilde{c}_{i,j} D_{\mathrm{HS}} \left[ \tilde{T}_i(\tilde{\sigma}^{(\ell+1)}), \tilde{T}_j(\tilde{\sigma}^{(\ell+1)}) \right], \quad \text{(B41)}$$

where we now remark that $\tilde{\sigma}_{w_{\mathrm{init}}}$ is the reduced state on the qubits that $W_{\mathrm{init}}$ acts on, instead of those that $W$ acts on. Equation (B41) leads to the following lower bound:

$$\int_{W_{\mathrm{init}}} \varepsilon_{\tilde{\sigma}_w} d\mu \geq \frac{1}{5} D_{\mathrm{HS}} \left( \tilde{\sigma}_{w_{\mathrm{init}}}^{(\ell+2)}, \frac{\mathbb{1}}{4} \mathrm{Tr}[\sigma] \right) \quad \text{(B42)}$$

$$= \frac{1}{5} \varepsilon_{\tilde{\sigma}_{w_{\mathrm{init}}}^{(\ell+2)}}. \quad \text{(B43)}$$

Now we have to integrate $(L - \ell)$ middle modules in $\tilde{\mathcal{L}}_B$, meaning that we can simply employ the result in Eq. (B40) to obtain

$$\langle \varepsilon_{\tilde{\sigma}_w} \rangle_{V_L} \geq \frac{1}{5} \left( \frac{1}{50} \right)^{L-\ell} \varepsilon_{\sigma_{w_{\mathrm{init}}}}. \quad \text{(B44)}$$

### d. W in the second sublayer but not in the edge

As shown in Fig. 13(b), here we consider the case when the unitary $W$ is in the second sublayer of the $\ell$th layer of the QCNN, but it is not the first or the last unitary. Similar to the previous case, here we have to integrate a unitary $W_{\mathrm{init}}$. We find

$$\int_{W_{\mathrm{init}}} \varepsilon_{\tilde{\sigma}_w} d\mu = \frac{1}{50} D_{\mathrm{HS}} \left( \tilde{\sigma}_{w_{\mathrm{init}}}^{(\ell+2)}, \frac{\mathbb{1}}{4} \mathrm{Tr}[\sigma] \right)$$
$$+ \sum_{i,j} \tilde{c}_{ij} D_{\mathrm{HS}} \left[ \tilde{T}_i(\tilde{\sigma}^{(\ell+1)}), \tilde{T}_j(\tilde{\sigma}^{(\ell+1)}) \right] \quad \text{(B45)}$$

leading to the following lower bound:

$$\int_{W_{\mathrm{init}}} \varepsilon_{\tilde{\sigma}_w} d\mu \geq \frac{1}{50} D_{\mathrm{HS}} \left( \tilde{\sigma}_w^{(1)}, \frac{\mathbb{1}}{4} \mathrm{Tr}[\sigma] \right) \quad \text{(B46)}$$

$$= \frac{1}{50} \varepsilon_{\tilde{\sigma}_{w_{\mathrm{init}}}^{(\ell+2)}}. \quad \text{(B47)}$$

Since the remaining unitaries can be grouped into $M_{\mathcal{M}}$ modules, we can use Eq. (B40) to obtain

$$\langle \varepsilon_{\tilde{\sigma}_w} \rangle_{V_L} \geq \frac{1}{50} \left( \frac{1}{50} \right)^{L-\ell} \varepsilon_{\sigma_{w_{\mathrm{init}}}}. \quad \text{(B48)}$$
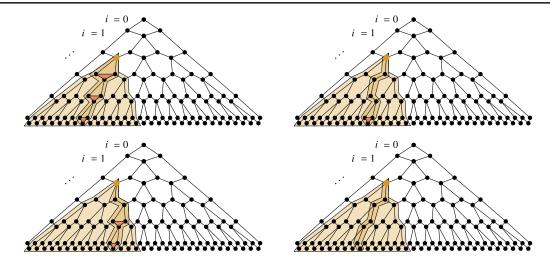
FIG. 14. By integrating a single unitary in each sublayer, one can obtain a reduced state $\hat{\sigma}$ in the subspace of any odd pair of qubits in $\mathcal{L}_B$.

### 4. General lower bound on the variance

Here we combine the results previously obtained to derive a general lower bound for $\mathrm{Var}[\partial_\mu C]$ which is independent of whether $W$ is in the first or in the second sublayer.

Let us start by noting that Eqs. (B40), (B44), and (B48) all provide a lower bound for the expectation value of $\langle \varepsilon_{\tilde{\sigma}_w} \rangle_{V_L}$. Specifically, the bound in Eqs. (B44) and (B48) can be obtained from that of Eq. (B40) by, respectively, dividing by $\frac{1}{5}$ and $\frac{1}{50}$. Taking the minimum of these three lower bounds, i.e., the one associated with the coefficient $\frac{1}{50}$, gives the general lower bound for the backward light cone:

$$\langle \varepsilon_{\tilde{\sigma}_w} \rangle_{V_L} \geq \left( \frac{1}{50} \right)^{L-\ell+1} \varepsilon_{\hat{\sigma}}, \qquad (B49)$$

where $\hat{\sigma} = \sigma_w$ if $W$ is in the first sublayer, or $\hat{\sigma} = \sigma_{w_{\text{init}}}$ if $W$ is in the second sublayer.

Combining Eqs. (B3), (B31), with (B49) results in a generalized form of Theorem 1

$$\mathrm{Var}[\partial_\mu C] \geq \frac{1}{9} \frac{\mathrm{Tr}[H_\mu^2] \varepsilon_O \varepsilon_{\hat{\sigma}}}{50^{L-\ell+2}} \sum_{\boldsymbol{g} \in P_\ell(\mathcal{G}_w)} \Lambda_{\boldsymbol{g}}. \qquad (B50)$$

Moreover, one can further generalize this result by noting that, as shown in Fig. 14, one can sequentially apply the trick of integrating a single $W_{ij}$ unitary in each sublayer to change how $\tilde{\mathcal{L}}_B$ is defined. Now, $\tilde{\mathcal{L}}_B$ can be composed of a single unitary per sublayer and then of a sequence of center modules. In this final effective light-cone, we can take $\hat{\sigma}$ to be reduced state in *any* odd pair of qubits in $\mathcal{L}_B$.

### APPENDIX C: PROOF OF COROLLARY 1

In this Appendix, we present the proof of Corollary 1, which requires us to analyze the scaling of the term $\sum_{\boldsymbol{g} \in P_\ell(\mathcal{G}_w)} \Lambda_{\boldsymbol{g}}$ in Theorem 1. Specifically, assuming that $L \in \mathcal{O}(\log(n))$ and that $\mathrm{Tr}[H_\mu^2] \varepsilon_O \varepsilon_{\hat{\sigma}} \in \Omega(1/\mathrm{poly}(n))$, we need to show that there is always one path $\boldsymbol{g}$ such that the coefficient $\Lambda_{\boldsymbol{g}}$ is not exponentially vanishing with the system size, i.e., that $\Lambda_{\boldsymbol{g}} \in \Omega(1/\mathrm{poly}(n))$. Moreover, we recall that we have to consider three different cases:

(i) Case 1: The light cone $\mathcal{L}_F$ can be covered by $\ell$ modules $M_{\mathcal{C}}$.

(ii) Case 2: The light cone $\mathcal{L}_F$ can be covered by $(\ell - 1)$ modules $M_{\mathcal{E}}$ and a module $M_{\mathcal{C}}$.

(iii) Case 3: The light cone $\mathcal{L}_F$ can be covered by $\ell'$ modules $M_{\mathcal{M}}$, $\ell''$ modules $M_{\mathcal{E}}$, and a module $M_{\mathcal{C}}$ such that $\ell' + \ell'' + 1 = \ell$.

### 1. Case 1

Let us recall that the graph for the case 1 (when $W$ is in the first sublayer) is presented in the main text. As shown in Fig. 15(a) (top), there is a single path through $\mathcal{G}_w$ and we have

$$\sum_{\boldsymbol{g} \in P_\ell(\mathcal{G}_w)} \Lambda_{\boldsymbol{g}} = \left( \frac{28}{125} \right)^\ell. \qquad (C1)$$

Then, recalling that $\ell \leq L$, it follows that

$$\sum_{\boldsymbol{g} \in P_\ell(\mathcal{G}_w)} \Lambda_{\boldsymbol{g}} \geq \left( \frac{28}{125} \right)^L, \qquad (C2)$$

where we see that the lower bound is in $\Omega(1/\mathrm{poly}(n))$ if $L \in \mathcal{O}(\log(n))$.
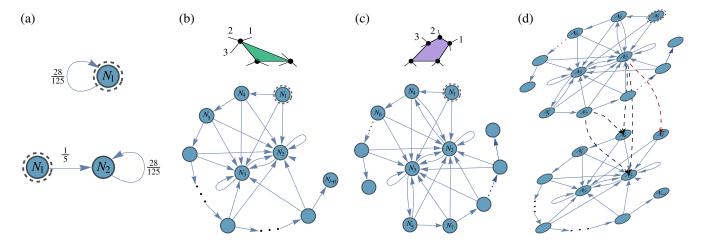
FIG. 15.   Graphs $\mathcal{G}_w$ obtained through the GRIM by integrating (a) $\ell$ center modules $M_\mathcal{C}$, (b) $\ell$ edge modules $M_\mathcal{E}$, (c) $\ell$ middle modules $M_\mathcal{M}$. In panels (b) and (c), we indicate how the wires are labeled. (d) Transition between the middle and the edge graphs of (b) and (c). All the nodes in the middle graph are connected to the nodes $N_2$ and $N_4$ on the edge graph. The node $N_2$ of the middle graph is also connected to a node $\hat{N}_1$ that connects to the rest of the edge graph.

The case when $W$ is in the second sublayer leads to an additional module being added to the graph [see Fig. 15(a), bottom]. This result simply modifies the lower bound as

$$\sum_{g \in P_\ell(\mathcal{G}_w)} \Lambda_g \geq \frac{1}{5} \left( \frac{28}{125} \right)^L, \qquad (C3)$$

and we recover again that the lower bound is in $\Omega(1/\text{poly}(n))$ if $L \in \mathcal{O}(\log(n))$.

### 2. Case 2

Let us first consider the case when $W$ is in the first sublayer. The graph obtained after integrating $\ell - 1$ modules is shown in Fig. 15(b). Here, let us describe the nodes and their associated coefficients.

The nodes in Fig. 15(b) are given by the contractions

$$N_1 = T_{\{\}} - \frac{1}{4} T_{\{1,2\}}, \qquad (C4)$$

$$N_2 = T_{\{\}} - \frac{1}{2} T_{\{1\}} - \frac{1}{2} T_{\{2,3\}} + \frac{1}{8} T_{\{1,2,3\}}, \qquad (C5)$$

$$N_3 = -T_{\{\}} + \frac{1}{2} T_{\{1\}} + 4 T_{\{2,3\}} - 2 T_{\{1,2,3\}}, \qquad (C6)$$

$$N_4 = -T_{\{\}} + 8 T_{\{1\}} + \frac{1}{2} T_{\{2,3\}} - 2 T_{\{1,2,3\}}, \qquad (C7)$$

$$N_5 = T_{\{\}} + 2 T_{\{1\}} - \frac{1}{2} T_{\{2,3\}} - \frac{1}{2} T_{\{1,2,3\}}, \qquad (C8)$$

$$N_{5+k} = T_{\{\}} + \frac{2(4-k)}{4k-1} T_{\{1\}} - \frac{1}{4} T_{\{2,3\}} - \frac{4-k}{2(4k-1)} T_{\{1,2,3\}}. \qquad (C9)$$

In Fig. 15(b), we show how the wires are labeled. Similarly, the edge coefficients $\lambda_{\alpha,\beta}$ are presented in Table I.

Finally, for case 2, the final module that one needs to integrate is a center module $M_\mathcal{C}$. Surprisingly, we find that for all the nodes $N_\alpha$ previously presented, we have

TABLE I.   Edge coefficients $\lambda_{\alpha,\beta}$ for the edge module graph.

| $\alpha$ | | | $\beta$ | | | |
|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | $5+k+1$ |
| 1 | 0 | $\frac{4}{375}$ | $\frac{2}{375}$ | $\frac{1}{75}$ | 0 | 0 |
| 2 | 0 | $\frac{24}{125}$ | $\frac{1}{25}$ | 0 | 0 | 0 |
| 3 | 0 | $\frac{32}{125}$ | $\frac{4}{25}$ | $\frac{4}{25}$ | 0 | 0 |
| 4 | 0 | $\frac{32}{25}$ | $\frac{4}{25}$ | 0 | $\frac{12}{25}$ | 0 |
| $5+k$ | 0 | $32(k+1)/125(4k-1)$ | $4(k+1)/125(4k-1)$ | 0 | 0 | $16(k+1)-4)/125(4k-1)$ |

$$\int_{M_{\mathcal{M}}} N_2(\tilde{O}_1)d\mu = \frac{72}{125}\varepsilon_O, \qquad (C10)$$

$$\int_{M_{\mathcal{C}}} N_3(\tilde{O}_1)d\mu = \frac{48}{125}\varepsilon_O, \qquad (C11)$$

$$\int_{M_{\mathcal{C}}} N_4(\tilde{O}_1)d\mu = \frac{528}{125}\varepsilon_O, \qquad (C12)$$

$$\int_{M_{\mathcal{C}}} N_5(\tilde{O}_1)d\mu = \frac{272}{125}\varepsilon_O, \qquad (C13)$$

$$\int_{M_{\mathcal{C}}} N_{5+k}(\tilde{O}_1)d\mu = \frac{528}{125(4k-1)}\varepsilon_O \quad \forall\ k \geq 1; \qquad (C14)$$

that is, all integrals generate positive coefficients.

From here, we can see that one can always find a path $\hat{g}$ such that $\Lambda_{\hat{g}}$ is a product of numbers $\lambda_i$ in $(0,1)$, meaning that

$$\sum_{g \in P_\ell(\mathcal{G}_w)} \Lambda_g \geq \Lambda_{\hat{g}} \qquad (C15)$$

$$= \prod_{i=1}^{L} \lambda_i \qquad (C16)$$

$$\geq (\lambda_{\min})^L, \qquad (C17)$$

where $\lambda_{\min}$ is the smallest $\lambda_i$ in the path $\hat{g}$. Hence, the lower bound is in $\Omega(1/\mathrm{poly}(n))$ if $L \in \mathcal{O}(\log(n))$.

When $L$ is in the second sublayer, we simply find that the graph in Fig. 15(b) is slightly modified, but the scaling of the lower bound for $\sum_{g \in P_\ell(\mathcal{G}_w)} \Lambda_g$ remains unchanged.

### 3. Case 3

Let us first consider the integration of the $\ell'$ middle modules $M_{\mathcal{M}}$. The graph obtained for the case when $W$ is in the first sublayer is presented in Fig. 15(c).

The nodes in Fig. 15(c) are given by the contractions

$$N_1 = T_{\{\}} - \frac{1}{4}T_{\{1,2\}}, \qquad (C18)$$

$$N_2 = -T_{\{\}} + \frac{1}{10}T_{\{1\}} + \frac{8}{5}T_{\{3\}} + \frac{1}{5}T_{\{1,2\}}$$
$$+ \frac{16}{5}T_{\{2,3\}} - 2T_{\{1,2,3\}}, \qquad (C19)$$

$$N_3 = T_{\{\}} - \frac{1}{10}T_{\{1\}} - \frac{1}{10}T_{\{3\}} - \frac{1}{5}T_{\{1,2\}} - \frac{1}{5}T_{\{2,3\}}$$
$$+ \frac{1}{8}T_{\{1,2,3\}}, \qquad (C20)$$

$$N_4 = -T_{\{\}} + \frac{8}{5}T_{\{1\}} + \frac{1}{10}T_{\{3\}} + \frac{16}{5}T_{\{1,2\}}$$
$$+ \frac{1}{5}T_{\{2,3\}} - 2T_{\{1,2,3\}}, \qquad (C21)$$

$$N_{2k+3} = T_{\{\}} + \frac{4(2b_k - a_k)}{5(8a_k - b_k)}T_{\{1\}} - \frac{1}{10}T_{\{3\}}$$
$$+ \frac{8(2b_k - a_k)}{5(8a_k - b_k)}T_{\{1,2\}} - \frac{1}{5}T_{\{2,3\}}$$
$$- \frac{2b_k - a_k}{8a_k - b_k}T_{\{1,2,3\}}, \quad \forall\ k \geq 1, \qquad (C22)$$

$$N_{2k+4} = T_{\{\}} + \frac{2(4a_k - b_k)}{5(4b_k - a_k)}T_{\{1\}} - \frac{1}{10}T_{\{3\}}$$
$$+ \frac{4(4a_k - b_k)}{5(4b_k - a_k)}T_{\{1,2\}} - \frac{1}{5}T_{\{2,3\}}$$
$$- \frac{4a_k - b_k}{8b_k - a_k}T_{\{1,2,3\}}, \quad \forall\ k \geq 1. \qquad (C23)$$

In Fig. 15(c), we show how the wires are labeled. The edge coefficients $\lambda_{\alpha,\beta}$ are presented in Table II, where the coefficients $a_k$, $b_k$, $p_k$, and $q_k$ follow a recursive formula of the form

$$f_n = 2f_{n-1} + f_{n-1}, \qquad (C24)$$

with

$$a_0 = 1, \qquad a_1 = 1, \qquad a_2 = 3, \qquad (C25)$$

$$b_0 = 0, \qquad b_1 = 2, \qquad b_2 = 4, \qquad (C26)$$

$$p_0 = 1, \qquad p_1 = 7, \qquad p_2 = 15, \qquad (C27)$$

$$q_0 = 1, \qquad q_1 = 4, \qquad q_2 = 11. \qquad (C28)$$

Hence, it is clear that $\lambda_{\alpha,\beta} \geq 0$ for all $\alpha$ and $\beta$.

Unlike case 2, here we have to also take into account that after integrating the $\ell'$ middle modules $M_{\mathcal{M}}$, one still needs to integrate $\ell''$ edge modules $M_{\mathcal{E}}$. As shown in Fig. 15(d), the result can be obtained by connecting the graph of Fig. 15(c) to a slightly modified edge graph from Fig. 15(b). Specifically, the edge graph needs to be modified by replacing the node $N_1$ by a modified node $\tilde{N}_1$ defined as

$$\tilde{N}_1 = T_{\{\}} - \frac{1}{2}T_{\{1\}} + \frac{92}{7}T_{\{2,3\}} - \frac{46}{7}T_{\{1,2,3\}}. \qquad (C29)$$

Note that $\tilde{N}_1$ is connected to the nodes $N_2$ and $N_3$ but not to the node $N_4$, meaning that we have to set $\lambda_{1,4} = 0$ in the edge graph. Then, as shown in Fig. 15(d), every node in the middle graph is connected to the nodes $N_2$ and $N_4$ on edge

TABLE II. Edge coefficients $\lambda_{\alpha,\beta}$ for the middle module graph and for $k \geq 1$.

| $\alpha$ | $\beta$ | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | $2k+5$ | $2k+6$ |
| 1 | 0 | $\frac{1}{750}$ | $\frac{56}{1875}$ | $\frac{1}{750}$ | 0 | 0 | 0 | 0 |
| 2 | 0 | $\frac{2}{125}$ | $\frac{288}{3125}$ | $\frac{8}{625}$ | 0 | 0 | 0 | 0 |
| 3 | 0 | $\frac{272}{3125}$ | $\frac{1}{250}$ | 0 | $\frac{6}{625}$ | 0 | 0 | 0 |
| 4 | 0 | $\frac{9}{250}$ | $\frac{488}{3125}$ | 0 | 0 | $\frac{56}{625}$ | 0 | 0 |
| $2k+3$ | 0 | $b_k + 4b_{k+1}/$ $250(8a_k - b_k)$ | $\frac{8}{3125}\{[8(17a_k + q_k)/8a_k - b_k]$ $+(25a_k/4b_k - a_k)$ $+(20a_{k+1}/4b_k - a_k)\}$ | 0 | 0 | | $8(8a_{k+1} - b_{k+1})/$ $625(8a_k - b_k)$ | 0 |
| $2k+4$ | 0 | $4a_{k+1} + a_k/$ $250(4b_k - a_k)$ | $\frac{8}{3125}(25a_k + 2500a_{k+1}$ $+68b_k + 16p_k/4b_k - a_k)$ | 0 | 0 | | 0 | $8(4b_{k+1} - a_{k+1})/$ $625(4b_k - a_k)$ |

graph. The only exception is the node $N_2$ of the middle graph which additionally is connected to the node $\hat{N}_1$. In Table III, we present the transition coefficients $\lambda_{\alpha,\beta}$, where the node $N_\alpha$ belongs to the middle graph, while the node $N_\beta$ to the edge graph. In addition, we also need present the coefficients $\lambda_{\tilde{1},\beta}$ for the updated $\tilde{N}_1$ *within* the edge graph: $\lambda_{\tilde{1},2} = 1168/875$ and $\lambda_{\tilde{1},3} = 33/175$. Finally, we remark that Eqs. (C10)–(C14) remain valid.

Similar to case 2, we can see that one can always find a path $\hat{g}$ such that $\Lambda_{\hat{g}}$ is a product of numbers $\lambda_i$ in $(0,1)$, meaning that

$$\sum_{g \in P_\ell(\mathcal{G}_w)} \Lambda_g \geq \Lambda_{\hat{g}} \tag{C30}$$

$$= \prod_{i=1}^{L} \lambda_i \tag{C31}$$

$$\geq (\lambda_{\min})^L, \tag{C32}$$

where $\lambda_{\min}$ is the smallest $\lambda_i$ in the path $\hat{g}$. Hence, the lower bound is in $\Omega(1/\text{poly}(n))$ if $L \in \mathcal{O}(\log(n))$. Moreover, a similar result can be obtained for $W$ in the second sublayer.

Hence, we show that Corollary 1 is valid for all possible choices of $W$.

## APPENDIX D: TRAINABILITY OF POOLING

By consolidating parameters into smaller registers, pooling layers in a QCNN can enhance trainability in comparison to a QNN in which a full $n$-qubit register is maintained throughout a variational circuit. This idea can be illustrated by the pooling module in Fig. 6, in which an input state $(\rho^{(0)})^{\otimes n}$ is processed by pooling layers locally described by the channel

$$\rho^{(j+1)} = \text{tr}_A[I_{j+1}(\rho^{(j)})^{\otimes 2}I_{j+1}^\dagger], \tag{D1}$$

where

$$I_{j+1} = |+\rangle\langle+|_A \otimes e^{-i\theta_+^{(j+1)}Y} + |-\rangle\langle-|_A \otimes e^{-i\theta_-^{(j+1)}Y} \tag{D2}$$

is the controlled unitary corresponding to the pooling. Starting from $\rho^{(0)} = |0\rangle^{\otimes n}$ with $n = 2^L$ for simplicity, a cost function at the $j$th layer ($j = 1, ..., L$) is given by

TABLE III. Transition coefficients $\lambda_{\alpha,\beta}$, where the node $N_\alpha$ belongs to the middle module graph, while the node $N_\beta$ belongs to the edge module graph.

| $\alpha$ | $\beta$ | | |
|---|---|---|---|
| | $\tilde{1}$ | 2 | 4 |
| 2 | $\frac{11}{125}$ | $\frac{64}{625}$ | $\frac{4}{625}$ |
| 3 | 0 | $\frac{136}{625}$ | $\frac{6}{625}$ |
| 4 | 0 | $\frac{64}{625}$ | $\frac{44}{625}$ |
| $2k+3$ | 0 | $\frac{64}{625}[(25a_k/8a_k - b_k) - 1]$ | $\frac{4}{625}[10(a_k + b_k) + b_{k+1}/8a_k - b_k]$ |
| $2k+4$ | 0 | $\frac{32}{625}(11b_k + 2p_k/4b_k - a_k)$ | $\frac{4}{625}(10a_k + a_{k+1} + 5b_k/4b_k - a_k)$ |

$$C^{(j)} = 1 - \text{Tr}[|0\rangle\langle0|^{\otimes(n/2^j)}(\rho^{(j)})^{\otimes(n/2^j)}]$$
$$= 1 - (\rho_{0,0}^{(j)})^{(n/2^j)}. \tag{D3}$$

The increase in trainability as $j \to L$ can be seen clearly by considering the cross section $\theta_{\pm}^{(j)} = \pm\theta$ for all $j$. For these angles, the cost function $C^{(j)}(\theta_+, \theta_-)$ is evaluated in closed form by solving the recursion relation

$$\rho_{0,0}^{(j+1)} - \rho_{1,1}^{(j+1)} = \cos\theta(\rho_{0,0}^{(j)} - \rho_{1,1}^{(j)}) \tag{D4}$$

subject to $\text{Tr}[\rho^{(j)}] = 1$ and $\rho_{0,0}^{(0)} = 1$ to obtain $\rho_{0,0}^{(j)} = (1 + \cos^j\theta/2)$. Substituting this into Eq. (D3) results in the closed form

$$C^{(j)}(\theta) = 1 - \left(\frac{1 + \cos^j\theta}{2}\right)^{\frac{n}{2^\ell}}. \tag{D5}$$

For $\ell = 1$, i.e., a single pooling layer, the expression (D5) is almost everywhere equal to 1 on $[-\pi, \pi]$ as $n \to \infty$ and, therefore, $C^{(1)}(\theta)$ exhibits a barren plateau. In contrast, for $j = \log n = L$, one evaluates

$$E\left(\left|\frac{dC^{(L)}}{d\theta}\right|\right) = \frac{L}{2}\int_{-\pi}^{\pi}\frac{d\theta}{2\pi}|\cos^{L-1}\theta\sin\theta|$$
$$= \frac{1}{\pi} \tag{D6}$$

for all $n$, which implies the absence of barren plateaus in the landscape because the magnitude of the derivative is bounded away from zero in expectation. Uncorrelating the pooling layers by taking the angles defining the conditional unitary $I_j$ to be $\theta_{\pm}^{(j)} := \pm\theta^{(j)}$, $j = 1, ..., L$ does not result in barren plateaus for any parameters, as long as the pooling module is sufficiently deep. In particular, the $j$ layer cost function

$$C^{(j)}(\{\theta^{(k)}\}_{k=1}^j) = 1 - \left(\frac{1 + \prod_{k=1}^j \cos\theta^{(k)}}{2}\right)^{\frac{n}{2^j}} \tag{D7}$$

for maximum depth $j = L$ satisfies

$$E\left(\left|\frac{dC^{(L)}}{d\theta^{(k)}}\right|\right) = \frac{1}{2n^{\log_2(\pi)-1}} \tag{D8}$$

for all $k$, which does not exhibit the exponentially fast decrease to 0 required by the definition of barren plateaus.

## APPENDIX E: GENERATION OF LOW-DEPTH TWO-QUBIT UNITARY 2-DESIGNS

In the proof of our main results, we assume that all the two-qubit unitary blocks present in the QCNN form a 2-design. We show here that this assumption can easily be fulfilled, even on near-term devices.
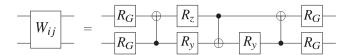


FIG. 16. Quantum circuit to implement an arbitrary unitary on two qubits. As shown in Ref. [63], $W_{ij}$ is universal (in that it can implement *any* two-qubit gate), and optimal (in that it contains the minimal number of CNOT gates and parameters, i.e., three CNOT gates and 15 parameters). Here, $R_G = e^{-iZ\theta_1/2}e^{-iY\theta_2/2}e^{-iZ\theta_3/2}$ is a general single-qubit rotation, and $R_\mu$ with $\mu \in \{y, z\}$ is a single-qubit rotation of the Bloch sphere about axis $\mu$.

In general, obtaining a unitary 2-design acting on $n$ qubits of a $D$-dimensional lattice requires a depth of $O(n^{1/D})$ [70–72]. However, for the QCNN architecture, one is interested only in 2-designs acting on two qubits. We show how to implement a 2-design without incurring any additional depth compared to a commonly used two-qubit variational ansatz.

First, let us recall that in a QCNN architecture, the convolutional layers are formed of unitaries acting on two-qubit gates. While the choice of these unitaries is *a priori* arbitrary, the most reasonable ansatz for these two-qubit unitaries is to employ the universal circuit shown in Fig. 16. As shown in Ref. [63], this circuit is universal (in that it can implement *any* two-qubit gate), and optimal (in that it contains the minimal number of CNOT gate and parameters, i.e., three CNOT gates and 15 parameters). Hence, any distribution of two-qubit gates (including a 2-design) can be efficiently implemented using the circuit in Fig. 16 without any additional depth. In fact, the distribution of unitaries obtained from the circuit in Fig. 16 solely depends on how the parameters in each gate are sampled, and not on the circuit depth. For instance, sampling all the angles uniformly in $[0, 2\pi]$ would not necessarily produce a 2-design.

However, it is possible to efficiently find a distribution over the angles that would generate Haar-distributed unitaries. For instance, this distribution can be obtained by generating random unitary matrices in $\mathcal{U}(4)$ using (for example) the algorithm described in Ref. [73] and compiling each unitary into the circuit of Fig. 16 using the quantum-assisted quantum-compiling technique of Ref. [19]. In this form, one can obtain a parameter distribution that forms a 2-design at no additional cost of circuit depth.

We remark that one could, in principle, also analyze the trainability of the QCNN with a different distribution of unitaries (or even with a simpler ansatz for the two-qubit unitaries). However, as we state in the main text, it is known that the most expressive circuits are the hardest to train [60], and hence, showing trainability for unitaries forming 2-designs indicates trainability for simpler cases. In this sense, our results constitute a "most random initialization scenario analysis" since we can provide rigorous trainability results for this presumed most random case.

[1] Y. LeCun, B. E. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard, and L. D. Jackel, in *Advances in Neural Information Processing Systems* (Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1990), pp. 396–404.

[2] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, *Gradient-Based Learning Applied to Document Recognition*, Proc. IEEE **86**, 2278 (1998).

[3] D. H. Hubel and T. N. Wiesel, *Receptive Fields and Functional Architecture of Monkey Striate Cortex*, J. Physiol. **195**, 215 (1968).

[4] K. Fukushima and S. Miyake, in *Competition and Cooperation in Neural Nets* (Springer, New York, 1982), pp. 267–285.

[5] W. Rawat and Z. Wang, *Deep Convolutional Neural Networks for Image Classification: A Comprehensive Review*, Neural Comput. **29**, 2352 (2017).

[6] N. Sharma, V. Jain, and A. Mishra, *An Analysis of Convolutional Neural Networks for Image Classification*, Proc. Comput. Sci. **132**, 377 (2018).

[7] A. A. M. Al-Saffar, H. Tao, and M. A. Talab, in *Proceedings of the 2017 International Conference on Radar, Antenna, Microwave, Electronics, and Telecommunications (ICRA-MET)* (IEEE, New York, 2017), pp. 26–31.

[8] S. Haykin, *Neural Networks: A Comprehensive Foundation* (Prentice-Hall, Englewood Cliffs, NJ, 1994).

[9] M. Minsky and S. A. Papert, *Perceptrons: An Introduction to Computational Geometry* (MIT Press, Cambridge, MA, 2017).

[10] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, *Learning Representations by Back-Propagating Errors*, Nature (London) **323**, 533 (1986).

[11] S. Hochreiter, Y. Bengio, P. Frasconi, J. Schmidhuber *et al.*, *Gradient Flow in Recurrent Nets: The Difficulty of Learning Long-Term Dependencies*, in *A Field Guide to Dynamical Recurrent Neural Networks*, edited by S. C. Kremer and J. F. Kolen (IEEE Press, 2001), https://www.bibsonomy.org/bibtex/279df6721c014a00bfac62abd7d5a9968/schaul.

[12] X. Glorot, A. Bordes, and Y. Bengio, *Deep Sparse Rectifier Neural Networks*, in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, Fort Lauderdale, FL, USA*, edited by G. Gordon, D. Dunson, and M. Dudík Proceedings of Machine Learning Research (PMLR, 2011), pp. 315–323, http://proceedings.mlr.press/v15/glorot11a/glorot11a.pdf.

[13] M. Schuld, I. Sinayskiy, and F. Petruccione, *An Introduction to Quantum Machine Learning*, Contemp. Phys. **56**, 172 (2015).

[14] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, *Quantum Machine Learning*, Nature (London) **549**, 195 (2017).

[15] W. Guan, G. Perdue, A. Pesah, M. Schuld, K. Terashi, S. Vallecorsa, and J.-R. Vlimant, *Quantum Machine Learning in High Energy Physics*, Mach. Learn. **2**, 011003 (2021).

[16] K. Sharma, M. Cerezo, Z. Holmes, L. Cincio, A. Sornborger, and P. J. Coles, *Reformulation of the No-Free-Lunch Theorem for Entangled Data Sets*, arXiv:2007.04900.

[17] A. Peruzzo, J. McClean, P. Shadbolt, M.-H. Yung, X.-Q. Zhou, P. J. Love, A. Aspuru-Guzik, and J. L. O'Brien, *A Variational Eigenvalue Solver on a Photonic Quantum Processor*, Nat. Commun. **5**, 4213 (2014).

[18] E. Farhi, J. Goldstone, and S. Gutmann, *A Quantum Approximate Optimization Algorithm*, arXiv:1411.4028.

[19] S. Khatri, R. LaRose, A. Poremba, L. Cincio, A. T. Sornborger, and P. J. Coles, *Quantum-Assisted Quantum Compiling*, Quantum **3**, 140 (2019).

[20] K. Sharma, S. Khatri, M. Cerezo, and P. J. Coles, *Noise Resilience of Variational Quantum Compiling*, New J. Phys. **22**, 043006 (2020).

[21] C. Bravo-Prieto, R. LaRose, M. Cerezo, Y. Subasi, L. Cincio, and P. J. Coles, *Variational Quantum Linear Solver: A Hybrid Algorithm for Linear Systems*, arXiv:1909.05820.

[22] C. Cirstoiu, Z. Holmes, J. Iosue, L. Cincio, P. J. Coles, and A. Sornborger, *Variational Fast Forwarding for Quantum Simulation beyond the Coherence Time*, npj Quantum Inf. **6**, 82 (2020).

[23] A. Arrasmith, L. Cincio, A. T. Sornborger, W. H. Zurek, and P. J. Coles, *Variational Consistent Histories as a Hybrid Algorithm for Quantum Foundations*, Nat. Commun. **10**, 3438 (2019).

[24] M. Cerezo, A. Poremba, L. Cincio, and P. J. Coles, *Variational Quantum Fidelity Estimation*, Quantum **4**, 248 (2020).

[25] M. Cerezo, A. Arrasmith, R. Babbush, S. C. Benjamin, S. Endo, K. Fujii, J. R. McClean, K. Mitarai, X. Yuan, L. Cincio, and P. J. Coles, *Variational Quantum Algorithms*, arXiv:2012.09265.

[26] K. Bharti, A. Cervera-Lierta, T. H. Kyaw, T. Haug, S. Alperin-Lea, A. Anand, M. Degroote, H. Heimonen, J. S. Kottmann, T. Menke *et al.*, *Noisy Intermediate-Scale Quantum (NISQ) Algorithms*, arXiv:2101.08448.

[27] M. Schuld, I. Sinayskiy, and F. Petruccione, *The Quest for a Quantum Neural Network*, Quantum Inf. Process. **13**, 2567 (2014).

[28] J. Romero, J. P. Olson, and A. Aspuru-Guzik, *Quantum Autoencoders for Efficient Compression of Quantum Data*, Quantum Sci. Technol. **2**, 045001 (2017).

[29] E. Farhi and H. Neven, *Classification with Quantum Neural Networks on Near Term Processors*, arXiv:1802.06002/

[30] N. Killoran, T. R. Bromley, J. M. Arrazola, M. Schuld, N. Quesada, and S. Lloyd, *Continuous-Variable Quantum Neural Networks*, Phys. Rev. Research **1**, 033063 (2019).

[31] G. Verdon, J. Marks, S. Nanda, S. Leichenauer, and J. Hidary, *Quantum Hamiltonian-Based Models and the Variational Quantum Thermalizer Algorithm*, arXiv:1910.02071.

[32] K. Beer, D. Bondarenko, T. Farrelly, T. J. Osborne, R. Salzmann, D. Scheiermann, and R. Wolf, *Training Deep Quantum Neural Networks*, Nat. Commun. **11**, 808 (2020).

[33] F. Tacchino, P. Barkoutsos, C. Macchiavello, I. Tavernelli, D. Gerace, and D. Bajoni, *Quantum Implementation of an Artificial Feed-Forward Neural Network*, Quantum Sci. Technol. **5**, 044010 (2020).

[34] I. Cong, S. Choi, and M. D. Lukin, *Quantum Convolutional Neural Networks*, Nat. Phys. **15**, 1273 (2019).

[35] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven, *Barren Plateaus in Quantum Neural Network Training Landscapes*, Nat. Commun. **9**, 4812 (2018).

[36] M. Cerezo, A. Sone, T. Volkoff, L. Cincio, and P. J. Coles, *Cost-Function-Dependent Barren Plateaus in Shallow Quantum Neural Networks*, Nat. Commun. **12**, 1791 (2021).

[37] K. Sharma, M. Cerezo, L. Cincio, and P. J. Coles, *Trainability of Dissipative Perceptron-Based Quantum Neural Networks*, arXiv:2005.12458.

[38] M. Cerezo and P. J. Coles, *Impact of Barren Plateaus on the Hessian and Higher Order Derivatives*, Quantum Sci. Technol. **6**, 035006 (2021).

[39] S. Wang, E. Fontana, M. Cerezo, K. Sharma, A. Sone, L. Cincio, and P. J. Coles, *Noise-Induced Barren Plateaus in Variational Quantum Algorithms*, arXiv:2007.14384.

[40] Z. Holmes, A. Arrasmith, B. Yan, P. J. Coles, A. Albrecht, and A. T. Sornborger, *Barren Plateaus Preclude Learning Scramblers*, Phys. Rev. Lett. **126**, 190501 (2021).

[41] T. J. Volkoff, *Efficient Trainability of Linear Optical Modules in Quantum Optical Neural Networks*, arXiv:2008.09173.

[42] E. Campos, A. Nasrallah, and J. Biamonte, *Abrupt Transitions in Variational Quantum Circuit Training*, Phys. Rev. A **103**, 032607 (2021).

[43] C. O. Marrero, M. Kieferová, and N. Wiebe, *Entanglement Induced Barren Plateaus*, arXiv:2010.15968.

[44] A. Abbas, D. Sutter, C. Zoufal, A. Lucchi, A. Figalli, and S. Woerner, *The Power of Quantum Neural Networks*, NATO ASI series Series F, Computer and system sciences **1**, 403 (2021).

[45] R. LaRose, A. Tikku, É. O'Neel-Judy, L. Cincio, and P. J. Coles, *Variational Quantum State Diagonalization*, npj Quantum Inf. **5**, 1 (2018).

[46] E. Grant, L. Wossnig, M. Ostaszewski, and M. Benedetti, *An Initialization Strategy for Addressing Barren Plateaus in Parametrized Quantum Circuits*, Quantum **3**, 214 (2019).

[47] T. Volkoff and P. J. Coles, *Large Gradients via Correlation in Random Parameterized Quantum Circuits*, Quantum Sci. Technol. **6**, 025008 (2021).

[48] G. Verdon, M. Broughton, J. R. McClean, K. J. Sung, R. Babbush, Z. Jiang, H. Neven, and M. Mohseni, *Learning to Learn with Quantum Neural Networks via Classical Neural Networks*, arXiv:1907.05415.

[49] A. Skolik, J. R. McClean, M. Mohseni, P. van der Smagt, and M. Leib, *Layerwise Learning for Quantum Neural Networks*, arXiv:2006.14904.

[50] M. Cerezo, K. Sharma, A. Arrasmith, and P. J. Coles, *Variational Quantum State Eigensolver*, arXiv:2004.01372.

[51] B. Commeau, M. Cerezo, Z. Holmes, L. Cincio, P. J. Coles, and A. Sornborger, *Variational Hamiltonian Diagonalization for Dynamical Quantum Simulation*, arXiv:2009.02559.

[52] K. Bharti and T. Haug, *Iterative Quantum Assisted Eigensolver*, arXiv:2010.05638.

[53] A. Cervera-Lierta, J. S. Kottmann, and A. Aspuru-Guzik, *The Meta-Variational Quantum Eigensolver (META-VQE): Learning Energy Profiles of Parameterized Hamiltonians for Quantum Simulation*, PRX Quantum **2**, 020329 (2021).

[54] T. L. Patti, K. Najafi, X. Gao, and S. F. Yelin, *Entanglement Devised Barren Plateau Mitigation*, Phys. Rev. Research **3**, 033090 (2021).

[55] E. Grant, M. Benedetti, S. Cao, A. Hallam, J. Lockhart, V. Stojevic, A. G. Green, and S. Severini, *Hierarchical Quantum Classifiers*, npj Quantum Inf. **4**, 65 (2018).

[56] I. MacCormack, C. Delaney, A. Galda, N. Aggarwal, and P. Narang, *Branching Quantum Convolutional Neural Networks*, arXiv:2012.14439.

[57] L. Franken and B. Georgiev, in *Proceedings of ESANN* (2020), http://publica.fraunhofer.de/documents/N-615541.html.

[58] K. Mitarai, M. Negoro, M. Kitagawa, and K. Fujii, *Quantum Circuit Learning*, Phys. Rev. A **98**, 032309 (2018).

[59] M. Schuld, V. Bergholm, C. Gogolin, J. Izaac, and N. Killoran, *Evaluating Analytic Gradients on Quantum Hardware*, Phys. Rev. A **99**, 032331 (2019).

[60] Z. Holmes, K. Sharma, M. Cerezo, and P. J. Coles, *Connecting Ansatz Expressibility to Gradient Magnitudes and Barren Olateaus*, arXiv:2101.02138.

[61] B. Collins and P. Śniady, *Integration with Respect to the Haar Measure on Unitary, Orthogonal and Symplectic Group*, Commun. Math. Phys. **264**, 773 (2006).

[62] Z. Puchała and J. Adam Miszczak, *Symbolic Integration with Respect to the Haar Measure on the Unitary Groups*, Bulletin of the Polish Academy of Sciences, Technical Sciences **65**, 21 (2017).

[63] F. Vatan and C. Williams, *Optimal Quantum Circuits for General Two-Qubit Gates*, Phys. Rev. A **69**, 032315 (2004).

[64] M. Broughton *et al.*, TENSORFLOW QUANTUM: *A Software Framework for Quantum Machine Learning*, arXiv:2003 .02989.

[65] K. Zhang, M.-H. Hsieh, L. Liu, and D. Tao, *Toward Trainability of Quantum Neural Networks*, arXiv:2011 .06258.

[66] M. Fukuda, R. König, and I. Nechita, *RTNI—A Symbolic Integrator for Haar-Random Tensor Networks*, J. Phys. A **52**, 425303 (2019).

[67] R. A. Horn and C. R. Johnson, *Matrix Analysis* (Cambridge University Press, Cambridge, England, 2012).

[68] P. J. Coles, M. Cerezo, and L.Cincio, *Strong Bound between Trace Distance and Hilbert-Schmidt Distance for Low-Rank States*, Phys. Rev. A **100**, 022103 (2019).

[69] M. Raıssouli and I. H. Jebril, *Various Proofs for the Decrease Monotonicity of the Schatten's Power Norm, Various Families of $\mathbb{R}^n$-Norms and Some Open Problems*, Int. J. Open Probl. Compt. Math. **3**, 164 (2010), http://eprints.umi.ac.ma/id/eprint/623.

[70] A. W. Harrow and R. A. Low, *Random Quantum Circuits Are Approximate 2-Designs*, Commun. Math. Phys. **291**, 257 (2009).

[71] F. G. S. L. Brandao, A. W. Harrow, and M. Horodecki, *Local Random Quantum Circuits Are Approximate Polynomial-Designs*, Commun. Math. Phys. **346**, 397 (2016).

[72] A. Harrow and S. Mehraban, *Approximate Unitary t-Designs by Short Random Quantum Circuits Using Nearest-Neighbor and Long-Range Gates*, arXiv:1809.06957.

[73] F. Mezzadri, *How to Generate Random Matrices from the Classical Compact Groups*, Not. Am. Math. Soc. **54**, 592 (2007), http://www.ams.org/notices/200705/fea-mezzadri-web.pdf.