Editors' Suggestion

# Estimating the gradient and higher-order derivatives on quantum hardware

Andrea Mari,[1,2] Thomas R. Bromley,[1] and Nathan Killoran[1]
[1]*Xanadu, Toronto, Ontario, Canada M5G 2C8*
[2]*Unitary Fund, Berkeley, California 94703, USA*

(Received 8 September 2020; accepted 13 November 2020; published 11 January 2021; corrected 11 March 2021)

For a large class of variational quantum circuits, we show how arbitrary-order derivatives can be analytically evaluated in terms of simple parameter-shift rules, i.e., by running the same circuit with different shifts of the parameters. As particular cases, we obtain parameter-shift rules for the Hessian of an expectation value and for the metric tensor of a variational state, both of which can be efficiently used to analytically implement second-order optimization algorithms on a quantum computer. We also consider the impact of statistical noise by studying the mean-square error of different derivative estimators. Some of the theoretical techniques for evaluating quantum derivatives are applied to their typical use case: the implementation of quantum optimizers. We find that the performance of different estimators and optimizers is intertwined with the values of different hyperparameters, such as the step size or the number of shots. Our findings are supported by several numerical and hardware experiments, including an experimental estimation of the Hessian of a simple variational circuit and an implementation of the Newton optimizer.

DOI: 10.1103/PhysRevA.103.012405

## I. INTRODUCTION

Many algorithms for near-term quantum computers are based on variational circuits [1–6], i.e., sequences of quantum gates depending on classical parameters that are recursively optimized for solving specific problems. Some of the most important examples are the variational quantum eigensolver (VQE) [1,7], the quantum approximate optimization algorithm (QAOA) [8], and all variational implementations of quantum machine learning algorithms [2,3,6,9]. In practice, all these cases share the same workflow: A quantum computer is used for measuring one or more expectation values which are classically combined into some cost function to be minimized. For example, the cost function could be the expectation value of some given observable (VQE, QAOA, etc.) or, as typical in quantum machine learning, some function which quantifies the distance between an ideal model and its variational approximation. To minimize the cost function, one can use different iterative methods which often require estimation of derivatives of expectation values with respect to the variational parameters of the circuit. The analytic gradient of a quantum circuit can be experimentally evaluated using the so-called parameter-shift rule [10–13], which allows implementation of all gradient-based optimizers such as gradient descent.

This work is structured in two main parts. In the first part, we explore a variety of derivative estimators for quantum circuits, seeking to understand and map out the effectiveness of these different methods. Our first result is a generalization of the analytic parameter-shift rule to derivatives of arbitrary orders including, as particular cases, the Hessian and the Fubini-Study metric tensor [14,15]. Moreover, we analyze the analytic and finite-difference methods of estimating derivatives from a statistical perspective, taking into account the uncertainty which is unavoidable in any quantum computation involving a finite number of measurement shots.

In the second part of this work, we apply the theoretical tools studied in the first part to implement several optimization methods which require the experimental estimation of the gradient and higher-order derivatives. Specifically we focus on the Newton optimizer and the quantum natural gradient optimizer. We also consider a diagonal approximation of the Newton optimizer (borrowed from classical machine learning [16]) which has a negligible overhead compared to gradient descent but nonetheless is sensitive to the curvature of the cost function with respect to individual parameters.

Our generalization of the parameter-shift rule is complementary to other approaches for evaluating derivatives of quantum circuits. In Ref. [17] it was shown how to evaluate the derivatives which are necessary to implement a VQE algorithm in a quantum chemistry scenario. In Ref. [13] a methodology for replacing direct measurements with indirect ones was proposed, including an experimentally feasible method for estimating the metric tensor. Very recently, different generalizations of the parameter-shift rule to arbitrary gates [18,19] have been proposed which in principle could be directly applied to extend the domain of applicability of our results to arbitrary gates. Moreover, it is worthwhile to mention that the same 0 and $\pm\pi/2$ shifts which naturally emerge in our analysis were also used in Ref. [20] for a direct minimization of the cost function with respect to single parameters.

This work is structured as follows. In Sec. II we set the notation and define the derivative tensor. In Sec. III we derive the higher-order parameter-shift rules. In Sec. IV the statistical noise associated with analytic and finite-difference estimators is studied. In Sec. V the implementation of second-order optimizers is considered within the context of parameter-shift rules. Section VI focuses on investigating our findings in simulation and on hardware. We first give practical examples of gradient and Hessian estimation and then we use these quantities to minimize the expectation value of a simple variational circuit.[1]

## II. SETTING

Most algorithms designed for near-term quantum computers are based on the variational optimization of quantum circuits [1–6]. A variational quantum circuit acting on $n$ qubits is a sequence of gates depending on a set of classical parameters $\boldsymbol{\theta} = (\theta_1, \theta_2, \dots, \theta_m)$ and producing a family of unitary operations $U(\boldsymbol{\theta})$. Typically, the circuit is applied to a fixed reference state $|0\rangle$ and the expectation value of a Hermitian observable $M$ is measured:

$$f(\boldsymbol{\theta}) = \langle 0|U(\boldsymbol{\theta})^\dagger M U(\boldsymbol{\theta})|0\rangle. \tag{1}$$

The expectation value given in Eq. (1) (or some cost function which depends on it) is usually optimized with respect to the parameters $\boldsymbol{\theta}$. For example, if $M$ is the Hamiltonian of a system, we can approximate the ground-state energy by minimizing $f(\boldsymbol{\theta})$. Many optimization methods require one to estimate the gradient at each iteration. For more advanced optimizers (e.g., Newton's method), one also needs the Hessian matrix or higher-order derivatives.

In this work we are interested in evaluating derivatives of an arbitrary order $d$, which we can cast as a tensor with $d$ indices $j_1, j_2, \dots, j_d$ whose elements are the real quantities

$$g_{j_1, j_2, \dots, j_d}(\boldsymbol{\theta}) = \frac{\partial^d f(\boldsymbol{\theta})}{\partial\theta_{j_1}\partial\theta_{j_2}\cdots\partial\theta_{j_d}}, \tag{2}$$

where the gradient and the Hessian correspond to the particular cases with $d = 1$ and $d = 2$, respectively. A typical structure of many variational circuits which can be executed by near-term quantum computers is

$$U(\boldsymbol{\theta}) = V_m U_m(\theta_m)\cdots V_2 U_2(\theta) V_1 U_1(\theta_1), \tag{3}$$

where $V_j$ are constant arbitrary circuits, while $U_j(\theta_j)$ are rotationlike gates, i.e., characterized by a generator $H_j$ such that $H_j^2 = \mathbb{1}$ (involutory matrix) and so

$$U_j(\theta_j) = e^{-(i/2)H_j\theta_j} = \cos(\theta_j/2)\mathbb{1} - i\sin(\theta_j/2)H_j. \tag{4}$$

For example, all single-qubit rotations belong to this class. More generally, $H_j$ can be any multiqubit tensor product of Pauli matrices.

Note that this class can be extended to general gates. A possible method is to decompose a gate into a product of rotationlike gates [18]. Another approach is to decompose an arbitrary generator $H_j$ into a linear combination of Pauli operators and then evaluating derivatives with respect to each term using the stochastic method recently proposed in Ref. [19].

[1]Source code for the numerics and experiments in this paper is available from [21].

## III. PARAMETER-SHIFT RULES

The class of variational quantum circuits specified by Eqs. (3) and (4) are commonly used and there exists a simple parameter-shift rule to evaluate their gradients [10–13]. This rule provides the gradient analytically by evaluating the circuit with fixed shifts of $\pi/2$ in the parameters $\boldsymbol{\theta}$. We begin this section by showing how the established gradient rule can be generalized to arbitrary parameter shifts and then extend our findings to higher-order derivatives such as the Hessian and Fubini-Study metric tensor.

### A. First-order derivatives: The gradient

From the previous identity, the unitary conjugation of an arbitrary operator $\hat{K}$ by $U_j(\theta_j)$ can always be reduced to the sum of three terms

$$\hat{K}(\theta_j) = U_j(\theta_j)^\dagger \hat{K} U_j(\theta_j) = \hat{A} + \hat{B}\cos(\theta_j) + \hat{C}\sin(\theta_j), \tag{5}$$

where $\hat{A}$, $\hat{B}$, and $\hat{C}$ are operators independent of $\theta_j$ and involving only $\hat{K}$ and $\hat{H}_j$. Moreover, from the standard trigonometric addition and subtraction identities, we can deduce that

$$\frac{d\cos(x)}{dx} = \frac{\cos(x+s) - \cos(x-s)}{2\sin(s)}, \tag{6}$$

$$\frac{d\sin(x)}{dx} = \frac{\sin(x+s) - \sin(x-s)}{2\sin(s)}, \tag{7}$$

which are valid for any $s \neq k\pi$, $k \in \mathbb{Z}$. Differentiating both sides of Eq. (5) and using Eqs. (6) and (7), we obtain a parameter-shift rule which is valid at the operator level:

$$\frac{\partial}{\partial\theta_j}\hat{K}(\theta_j) = \frac{\hat{K}(\theta_j + s) - \hat{K}(\theta_j - s)}{2\sin(s)}. \tag{8}$$

Note that, even if the preceding expression looks like a finite-difference approximation, it is actually exact and we can use it to analytically estimate the gradient of expectation values whenever the rotationlike property of Eq. (4) holds. Indeed, the operator identity in Eq. (8) can be directly applied into Eq. (1) to evaluate the $j$th component of the gradient. The result is a family of parameter-shift rules

$$g_j(\boldsymbol{\theta}) = \frac{f(\boldsymbol{\theta} + s\mathbf{e}_j) - f(\boldsymbol{\theta} - s\mathbf{e}_j)}{2\sin(s)}, \tag{9}$$

where $\mathbf{e}_j$ is the unit vector along the $\theta_j$ axis.

Note that, in the limit $s \to 0$, $\sin(s)$ can be approximated by $s$ and we recover the central-difference approximation for the first derivative. On the other hand, for $s = \pi/2$ we obtain the parameter-shift rule already studied in [10–13] and used in quantum software libraries [22–26]. It is important to remark that the formula in Eq. (9) is exact for any choice of $s$. Strictly speaking, $s$ cannot be a multiple of $\pi$ because of the diverging denominator; however, all the corresponding discontinuities are removable.

### B. Second-order derivatives: The Hessian

A useful property of Eq. (9) is that it can be iterated to get higher-order derivatives. Applying the same rule twice, we get a similar formula for the Hessian,

$$g_{j_1,j_2}(\boldsymbol{\theta}) = \big[f\big(\boldsymbol{\theta}+s_1\mathbf{e}_{j_1}+s_2\mathbf{e}_{j_2}\big) - f\big(\boldsymbol{\theta}-s_1\mathbf{e}_{j_1}+s_2\mathbf{e}_{j_2}\big) - f\big(\boldsymbol{\theta}+s_1\mathbf{e}_{j_1}-s_2\mathbf{e}_{j_2}\big) + f\big(\boldsymbol{\theta}-s_1\mathbf{e}_{j_1}-s_2\mathbf{e}_{j_2}\big)\big] \times [4\sin(s_1)\sin(s_2)]^{-1}, \tag{10}$$

which, for $s_1 = s_2 = s$, simplifies to

$$g_{j_1,j_2}(\boldsymbol{\theta}) = \big[f\big(\boldsymbol{\theta}+s\big(\mathbf{e}_{j_1}+\mathbf{e}_{j_2}\big)\big) - f\big(\boldsymbol{\theta}+s\big(-\mathbf{e}_{j_1}+\mathbf{e}_{j_2}\big)\big) - f\big(\boldsymbol{\theta}+s\big(\mathbf{e}_{j_1}-\mathbf{e}_{j_2}\big)\big) + f\big(\boldsymbol{\theta}-s\big(\mathbf{e}_{j_1}+\mathbf{e}_{j_2}\big)\big)\big] \times [2\sin(s)]^{-2}. \tag{11}$$

Also in this case, for $s \to 0$, we get the standard central-difference formula for the Hessian. For $s = \pi/2$, we get an analytic parameter-shift rule which is similar to the gradient formula used in Refs. [10–13], but extended to the Hessian. A formula equivalent to Eq. (11) for the particular case $s = \pi/2$ was recently used in Refs. [17,27]. Particular attention should be paid to the diagonal of the Hessian since for each element two shifts are applied to the same parameter $\theta_j$. In this case, two alternative choices for the value of $s$ which appears in Eq. (11) are particularly relevant. For the choice $s = \pi/2$ we get

$$g_{j,j}(\boldsymbol{\theta}) = [f(\boldsymbol{\theta}+\pi\mathbf{e}_j) - f(\boldsymbol{\theta})]/2, \tag{12}$$

where we used that $f(\boldsymbol{\theta}+\pi\mathbf{e}_j) = f(\boldsymbol{\theta}-\pi\mathbf{e}_j)$. Instead, for $s = \pi/4$, we obtain

$$g_{j,j}(\boldsymbol{\theta}) = [f(\boldsymbol{\theta}+\mathbf{e}_j\pi/2) - 2f(\boldsymbol{\theta}) + f(\boldsymbol{\theta}-\mathbf{e}_j\pi/2)]/2. \tag{13}$$

Each of the two preceding formulas has alternative advantages. The advantage of Eq. (12) is that it involves only two expectation values and so it is more direct with respect to Eq. (13), which is instead a linear combination of three terms. On the other hand, the parameter shifts involved in Eq. (13) are only $\pm\pi/2$. This implies that all the elements of the full Hessian matrix can be evaluated using only the same type of $\pm\pi/2$ shifts and this fact could be an experimentally relevant simplification. Moreover, in the typical scenario in which one has already evaluated the gradient using the $m$ pairs of shifts $f(\boldsymbol{\theta} \pm \pi/2\mathbf{e}_j)$, Eq. (13) allows us to evaluate the diagonal of the Hessian with the extra cost of just a single expectation value, i.e., $f(\boldsymbol{\theta})$. In Sec. V we show how this fact can be conveniently exploited to replace the vanilla gradient descent optimizer with a diagonal approximation of the Newton optimizer, with negligible computational overhead.

### C. Fubini-Study metric tensor

A second-order tensor which plays an important role in quantum information theory is the Fubini-Study metric tensor which for a pure variational state $|\psi(\boldsymbol{\theta})\rangle$ can be expressed as

$$F_{j_1,j_2}(\boldsymbol{\theta}) = -\frac{1}{2}\frac{\partial^2}{\partial\theta_{j_1}\partial\theta_{j_2}}|\langle\psi(\boldsymbol{\theta}')|\psi(\boldsymbol{\theta})\rangle|^2\bigg|_{\theta'=\theta} \tag{14}$$

and corresponds to the real part of the quantum geometric tensor (see, e.g., Appendix A1 of [28] for a detailed derivation). For pure states and up to constant factors, Eq. (14) can also be associated with other tensors such as the quantum Fisher information matrix or the Bures metric tensor [15]. Since in this work we only deal with pure states, we often refer to Eq. (14) simply as the metric tensor.

Different from the Hessian, the metric tensor is not linked to a particular observable $M$ but is instead a geometric property of a variational quantum state, which in our setting is simply $|\psi(\boldsymbol{\theta})\rangle = U(\boldsymbol{\theta})|0\rangle$. This tensor plays a crucial role in the implementation of the quantum natural gradient optimizer [28] and in the variational quantum simulation of imaginary-time evolution [29].

Now we can make a useful observation: the metric tensor in Eq. (14) can actually be seen (up to a constant factor) as the Hessian of the expectation value $f(\boldsymbol{\theta})$ defined in Eq. (1), for the particular observable $M(\boldsymbol{\theta}') = U(\boldsymbol{\theta}')|0\rangle\langle 0|U(\boldsymbol{\theta}')$. Therefore, all the previous theoretical machinery that we have derived for the Hessian applies also to the metric tensor and we get the corresponding parameter-shift rule which is simply the same as Eq. (11), where each expectation value is

$$f(\boldsymbol{\theta}) = |\langle\psi(\boldsymbol{\theta}')|\psi(\boldsymbol{\theta})\rangle|^2. \tag{15}$$

The quantity $|\langle\psi(\boldsymbol{\theta}')|\psi(\boldsymbol{\theta})\rangle|^2$ is the survival probability of the state $|0\rangle$ after the application of the circuit $U(\boldsymbol{\theta}')U(\boldsymbol{\theta})$. This probability can be easily estimated with near-term quantum computers either with a SWAP test or more simply as the probability of obtaining the $00\ldots0$ bit string after measuring the state $U(\boldsymbol{\theta}')U(\boldsymbol{\theta})|0\rangle$ in the computational basis.

Substituting Eq. (15) into Eq. (11) and setting $s = \pi/2$ and $\theta' = \theta$, we get the explicit parameter-shift rule for the metric tensor:

$$F_{j_1,j_2}(\boldsymbol{\theta}) = -\tfrac{1}{8}\Big[\big|\langle\psi(\boldsymbol{\theta})|\psi\big(\boldsymbol{\theta}+\big(\mathbf{e}_{j_1}+\mathbf{e}_{j_2}\big)\pi/2\big)\rangle\big|^2 - \big|\langle\psi(\boldsymbol{\theta})|\psi\big(\boldsymbol{\theta}+\big(\mathbf{e}_{j_1}-\mathbf{e}_{j_2}\big)\pi/2\big)\rangle\big|^2 - \big|\langle\psi(\boldsymbol{\theta})|\psi\big(\boldsymbol{\theta}+\big(-\mathbf{e}_{j_1}+\mathbf{e}_{j_2}\big)\pi/2\big)\rangle\big|^2 + \big|\langle\psi(\boldsymbol{\theta})|\psi\big(\boldsymbol{\theta}-\big(\mathbf{e}_{j_1}+\mathbf{e}_{j_2}\big)\pi/2\big)\rangle\big|^2\Big]. \tag{16}$$

As we discussed for the Hessian, also in this case the formula for the diagonal elements can be simplified in two alternative ways. The first formula, corresponding to Eq. (12), is

$$F_{j,j}(\boldsymbol{\theta}) = \tfrac{1}{4}[1 - |\langle\psi(\boldsymbol{\theta})|\psi(\boldsymbol{\theta}+\pi\mathbf{e}_j)\rangle|^2]. \tag{17}$$

The second equivalent formula, corresponding to Eq. (13), is

$$F_{j,j}(\boldsymbol{\theta}) = \tfrac{1}{2}[1 - |\langle\psi(\boldsymbol{\theta})|\psi(\boldsymbol{\theta}+\mathbf{e}_j\pi/2)\rangle|^2], \tag{18}$$

where we used that $|\langle\psi(\boldsymbol{\theta})|\psi(\boldsymbol{\theta}+\mathbf{e}_j\pi/2)\rangle|^2 = |\langle\psi(\boldsymbol{\theta})|\psi(\boldsymbol{\theta}-\mathbf{e}_j\pi/2)\rangle|^2$. We also comment that an efficient method for evaluating diagonal blocks of the metric tensor was proposed in [28]. Moreover, an experimentally feasible methodology for measuring the metric tensor was proposed in [13].

### D. Arbitrary-order derivatives

The same iterative approach can be used to evaluate derivatives of arbitrary order. In this case, for simplicity, we set $s = \pi/2$ and we introduce the multiparameter shift vectors

$$\mathbf{k}_{\pm j_1,\pm j_2,\ldots,\pm j_d} = \frac{\pi}{2}\big(\pm\mathbf{e}_{j_1} \pm \mathbf{e}_{j_2} \pm \cdots \pm \mathbf{e}_{j_d}\big), \tag{19}$$

where $j_1, j_2, \ldots, j_d$ are the same $d$ indices which appear also in the derivative tensor defined in Eq. (2). These vectors represent all the shifts in parameter space that are generated by iterating the parameter-shift rule of Eq. (9) for $j \in \{j_1, j_2, \dots, j_d\}$. We explicitly introduce the set of all shifts which are related to a derivative of order $d$:

$$S_{j_1,j_2,\dots,j_d} = \{\mathbf{k}_{\pm j_1,\pm j_2,\dots,\pm j_d}\ \forall \text{ choices of signs}\}. \tag{20}$$

For example, for the Hessian formula given in Eq. (11) evaluated as $s = \pi/2$, there are four possible shifts: $S_{j_1,j_2} = \{\mathbf{k}_{\pm j_1,\pm j_2}\}$. With this notation, the derivative tensor of order $d$ defined in Eq. (2) can be expressed in terms of the generalized parameter-shift rule

$$g_{j_1,j_2,\dots,j_d}(\boldsymbol{\theta}) = \frac{1}{2^d} \sum_{\mathbf{k}\in S_{j_1,j_2,\dots,j_d}} \mathcal{P}(\mathbf{k}) f(\boldsymbol{\theta}+\mathbf{k}), \tag{21}$$

where $\mathcal{P}(\mathbf{k})$ is the parity of a shift vector, i.e., the parity of negative indices in Eq. (19).

How many expectation values are necessary to evaluate the $d$-order derivative defined in Eq. (2)? This is given by the number of elements in the previous sum, which is $|S_{j_1,j_2,\dots,j_d}| = 2^d$. When some of the indices are repeated, the number of shifts can be further reduced, but we neglect this fact for the moment. Taking into account that the derivative tensor is symmetric with respect to permutations of the indices, the number of distinct elements of a tensor of order $d$ is given by the combinations of $d$ indices sampled with replacement from the set of $m$ variational parameters, corresponding to the multiset coefficient $\binom{m+d-1}{d}$. Therefore, the total number of expectation values to evaluate a derivative tensor of order $d$ is bounded by

$$\text{No. of expectation values} \leqslant 2^d \binom{m+d-1}{d} = O(m^d), \tag{22}$$

where in the last step we assumed $m \gg d$. In the opposite regime $d \gg m$, each angle parameter $\theta_j$ can be subject to multiple shifts. However, because of the periodicity of $f(\boldsymbol{\theta})$, for each $\theta_j$ we have at most four shifts: $0, \pm\pi/2, \pi$. Moreover, from Eq. (5) it is easy to check that a $\pi$ shift can be expressed as a linear combination of the others,

$$f(\boldsymbol{\theta} \pm \pi \mathbf{e}_j) = f(\boldsymbol{\theta} + \mathbf{e}_j\pi/2) + f(\boldsymbol{\theta} - \mathbf{e}_j\pi/2) - f(\boldsymbol{\theta}), \tag{23}$$

which means that only three shifts for each parameter are actually enough: the trivial 0 shift and the two $\pm\pi/2$ shifts.

From this simple counting argument we get another upper bound

$$\text{No. of expectation values} \leqslant 3^m, \tag{24}$$

which is valid for any $d$. This implies an interesting fact: From a finite number of $3^m$ combinations of variational parameters $\boldsymbol{\theta}$ we can Taylor expand $f(\boldsymbol{\theta})$ up to an arbitrary order and so we can actually evaluate $f(\boldsymbol{\theta})$ for all possible values of $\boldsymbol{\theta}$.

This fact may seem quite surprising; however, it simply reflects the trigonometric structure of rotationlike gates. Indeed, from Eq. (5) we see that $\hat{K}(\theta_j)$ is a linear combination of three terms. Similarly, a full circuit with $m$ parameters can be expressed as a linear combination of $3^m$ operators. Therefore, a generic expectation value $f(\boldsymbol{\theta})$ is a linear combination of different trigonometric functions weighted by $3^m$ scalar coefficients, a fact which was already noticed in Ref. [20]. In principle, these coefficients could be fully determined by evaluating $f(\boldsymbol{\theta})$ at $3^m$ different values of $\boldsymbol{\theta}$. In this work we focus on the different combinations of the three specific shifts of $0, \pm\pi/2$ for each angle $\theta_j$. However, one could also obtain similar results using any other choice of three nontrivial shifts for each angle.

This concludes the first part of this work, in which we derived several exact analytical results. However, if we really want to apply Eqs. (9), (11), and (21) in a quantum experiment, we have to take into account that each expectation value on the right-hand side of these equations can be measured only up to a finite precision and so the derivative on the left-hand side can be estimated only up to a finite error.

## IV. STATISTICAL ESTIMATION OF DERIVATIVES

The expectation value in Eq. (1) is a theoretical quantity. In a real quantum computation with $N$ measurement shots, we can only estimate $f(\boldsymbol{\theta})$ up to a finite statistical uncertainty, i.e., what we actually measure is

$$\hat{f}(\boldsymbol{\theta}) = f(\boldsymbol{\theta}) + \hat{\epsilon}, \tag{25}$$

with $\hat{\epsilon}$ a zero-mean random variable with variance $\sigma^2 = \sigma_0^2/N$, where $\sigma_0^2$ is the variance for a single shot which depends on the specific details of the circuit and of the observable. The aim of this section is to take into account the statistical noise which appears in Eq. (25). We consider this problem within the theoretical framework of statistical inference and we study different derivative estimators for the quantities defined in Eq. (2). In particular, we are interested in two main classes of estimators: one based on a finite-shot version of the analytic parameter-shift rules derived in the preceding section and one based on a standard finite-difference approximation. More precisely, the analytic estimator for a derivative tensor of arbitrary order $d$ is

$$\hat{g}^{(s=\pi/2)}_{j_1,j_2,\dots,j_d} = \frac{1}{2^d} \sum_{\mathbf{k}\in S_{j_1,j_2,\dots,j_d}} \mathcal{P}(\mathbf{k}) \hat{f}(\boldsymbol{\theta}+\mathbf{k}), \tag{26}$$

where we simply placed Eq. (25) into (21).

We can already make a preliminary comparison by considering the central-difference approximation which has a very similar structure but involves an infinitesimal step size $h$ instead of $s = \pi/2$. In order to simplify the comparison we express it using the same notation introduced in Eqs. (19) and (20):

$$\hat{g}^{(h)}_{j_1,j_2,\dots,j_d} = \frac{1}{(2h)^d} \sum_{\mathbf{k}\in S_{j_1,j_2,\dots,j_d}} \mathcal{P}(\mathbf{k}) \hat{f}(\boldsymbol{\theta}+\mathbf{k}2h/\pi). \tag{27}$$

It is evident that the only difference introduced in Eq. (27), compared to (26), is that shifts have a step size $h$ and that the full estimator is scaled by $h^{-d}$. This directly implies that, for $h < 1$, the statistical variance of the central-difference estimator is amplified by a factor of $h^{-2d}$, which is a strong limitation with respect to the analytic estimator. A more detailed analysis of the statistical error characterizing the analytic and the finite-difference methods will be presented later for the particular case of the gradient ($d = 1$).

### A. Quantifying the error of a statistical estimator

In general terms, our aim is to find a good estimator $\hat{g}_{j_1,\dots,j_d}(\boldsymbol{\theta})$ which only depends on a finite number of measurement shots and which should approximate the derivative tensor $g_{j_1,\dots,j_d}(\boldsymbol{\theta})$ defined in Eq. (2) well. As a figure of merit for the performance of an estimator we can use its mean-square error (MSE) with respect to the true value, which is

$$\Delta(\hat{g}_{j_1,\dots,j_d}) = \mathbb{E}\big[\big(\hat{g}_{j_1,\dots,j_d} - g_{j_1,\dots,j_d}\big)^2\big], \tag{28}$$

where $\mathbb{E}(\cdot)$ is the (classical) average over the statistical distribution of the measurement outcomes. The performance of the estimator can also be measured with respect to the full tensor in terms of the total error,

$$\Delta(\hat{\boldsymbol{g}}) = \mathbb{E}[\|\hat{\boldsymbol{g}} - \boldsymbol{g}\|^2] = \sum_{j_1,\dots,j_d} \Delta\big(\hat{g}_{j_1,\dots,j_d}\big), \tag{29}$$

which is simply the sum of the single-element errors.

Before considering specific derivative estimators, it is useful to recall also the notions of bias and variance:

$$\mathrm{Bias}\big(\hat{g}_{j_1,\dots,j_d}\big) := \mathbb{E}\big(\hat{g}_{j_1,\dots,j_d}\big) - g_{j_1,\dots,j_d}, \tag{30}$$

$$\mathrm{Var}\big(\hat{g}_{j_1,\dots,j_d}\big) := \mathbb{E}\big(\hat{g}^2_{j_1,\dots,j_d}\big) - \mathbb{E}\big(\hat{g}_{j_1,\dots,j_d}\big)^2. \tag{31}$$

These two quantities correspond to different errors: The bias represents a constant error which remains present even in the limit of an infinite number of shots $N \to \infty$, while the variance represents statistical fluctuations which are due to a finite $N$. It is well known that both effects can increase the MSE, and indeed we have

$$\Delta\big(\hat{g}_{j_1,\dots,j_d}\big) = \mathrm{Var}\big(\hat{g}_{j_1,\dots,j_d}\big) + \mathrm{Bias}\big(\hat{g}_{j_1,\dots,j_d}\big)^2. \tag{32}$$

In the following sections we focus on the estimation of the gradient, but a similar analysis can be extended to the Hessian and higher-order derivatives.

### B. Finite-difference gradient estimator

A general way of approximating the gradient is to use a finite-difference estimator, which requires one to experimentally measure expectation values for slightly different values of the parameters. The most common finite-difference estimators are the forward-difference and the central-difference estimators, both well studied in classical numerical analysis [30]. In this work we focus mainly on the central-difference estimator because it has the same structure as the parameter-shift rules derived in the previous section and so we can easily make a comparison.

Given a fixed step size $h > 0$, the symmetric finite-difference estimator for the $j$th element of the gradient can be defined as

$$\begin{aligned}\hat{g}^{(h)}_j &= \frac{\hat{f}(\theta_j + h) - \hat{f}(\theta_j - h)}{2h}\\ &= \frac{f(\theta_j + h) - f(\theta_j - h)}{2h} + \frac{\hat{\epsilon}_+ - \hat{\epsilon}_-}{2h},\end{aligned} \tag{33}$$

where we used Eq. (25) and $\hat{\epsilon}_\pm$ is the statistical noise associated with $\hat{f}(\theta_j \pm h)$. In the right-hand side of Eq. (33) we have the sum of two terms which are the finite-difference approximation and the statistical noise, respectively. Each term introduces a different kind of error: One is linked to the finite step size $h$ and the other is linked to the finite number of shots $N$. Such errors correspond to the bias and to the variance of the estimator discussed in the preceding section. Indeed, from Eqs. (30) and (31) we have

$$\begin{aligned}\mathrm{Bias}\big(\hat{g}^{(h)}_j\big) &= \frac{f(\theta_j + h) - f(\theta_j - h)}{2h} - g_j\\ &= \frac{f_3 h^2}{3!} + O(h^3),\end{aligned} \tag{34}$$

$$\mathrm{Var}\big(\hat{g}^{(h)}_j\big) = \frac{\sigma_0^2(\theta_j + h) + \sigma_0^2(\theta_j - h)}{4Nh^2} \tag{35}$$

$$\approx \frac{\sigma_0^2}{2Nh^2}, \tag{36}$$

where, in Eq. (34), we used the Taylor-series approximation with $f_3 = \partial^3 f(\theta_j)/\partial\theta_j^3$ and $\sigma_0^2(\theta_j \pm h)$ is the single-shot variance evaluated at $\theta_j \pm h$. The step from Eq. (35) to Eq. (36) is justified only if the following assumption holds.

*Assumption 1.* The variance of the measured observable depends weakly on the parameter shift such that $\sigma_0^2(\theta_j + x) + \sigma_0^2(\theta_j - x) \simeq 2\sigma_0^2$ for any value of $x$.

This assumption is usually a quite good approximation; however, one can easily construct counterexamples in which this is violated for large values of the shift. For this reason, whenever a subsequent result depends on Assumption 1, it will be explicitly stressed.

It is evident that the terms in Eqs. (34) and (36) have opposite scaling with respect to the step size: For small $h$ the variance diverges, while for large $h$ the bias dominates. This trade-off implies that there must exist an optimal choice of $h$.

Substituting Eqs. (34) and (36) into Eq. (32) we get, up to $O(h^6)$ corrections and within the validity of Assumption 1, the MSE for an arbitrary step size

$$\Delta\big(\hat{g}^{(h)}_j\big) \simeq \frac{\sigma_0^2}{2Nh^2} + \frac{f_3^2 h^4}{36}. \tag{37}$$

Imposing the derivative with respect to $h$ to be zero and assuming $f_3 \neq 0$, we get the optimal step size $h^*$ and the optimal error

$$h^* = \left(\frac{9\sigma_0^2}{f_3^2 N}\right)^{1/6} \propto N^{-1/6} \simeq N^{-0.167}, \tag{38}$$

$$\begin{aligned}\Delta\big(\hat{g}^{(h^*)}_j\big) &= \frac{3}{2}\frac{\sigma_0^2}{2N}(h^*)^{-2} = \frac{3}{2}\left[\frac{\sigma_0^2}{2N}\right]^{2/3}\left[\frac{f_3^2}{18}\right]^{1/3}\\ &\propto N^{-2/3} \simeq N^{-0.667},\end{aligned} \tag{39}$$

which are valid only for sufficiently large $N$ (i.e., for sufficiently small $h^*$). When the number of shots is small, the optimal step $h^*$ can become so large that both the Taylor approximation and Assumption 1 may not be valid anymore, so we could observe deviations from the predictions of Eqs. (37)–(39).

From a practical point of view, it is not straightforward to determine the optimal step $h^*$ since it depends on the third derivative $f_3$. The situation is significantly simpler for rotationlike gates where, from Eq. (5), we have that the third derivative is equal to the first and so $f_3$ can be replaced by $g_j$. However, as we are going to see in the following sections, in this case it is more convenient to use the parameter-shift estimator. For the sake of completeness we comment that, by repeating the same analysis for the forward-difference estimator $\hat{G}_j^{(h)} = [\hat{f}(\theta_j + h) - \hat{f}(\theta_j)]/h$, one gets similar scaling laws but with different exponents,

$$\Delta\big(\hat{G}_j^{(h)}\big) \simeq \frac{\sigma_0^2}{2Nh^2} + \frac{f_2^2 h^2}{4}, \tag{40}$$

$$h^* = \left(\frac{2\sigma_0^2}{f_2^2 N}\right)^{1/4} \propto N^{-0.25}, \tag{41}$$

$$\Delta\big(\hat{G}_j^{(h^*)}\big) = \frac{\sigma_0^2}{N}(h^*)^{-2} = \left(\frac{\sigma_0^2 f_2^2}{2N}\right)^{1/2} \propto N^{-0.5}, \tag{42}$$

which are valid as long as the approximations in the Taylor truncation and in Assumption 1 are justified.

It is notable that a similar trade-off for the choice of the step size $h$ was studied also in the field of classical numerical analysis (see, e.g., [30,31]). In a classical computer the error term $\hat{\epsilon}$ in Eqs. (25) and (33) is the roundoff error (or machine epsilon) associated with the finite-precision representation of real numbers. Optimizing the value of $h$ was particularly useful in the early days of classical computing, because of the limited memory and computational resources of that time. Nowadays, with current quantum computers, we are in a similar situation: The statistical uncertainty of quantum measurements and the limited number measurement shots give rise to a quantum roundoff error, which is usually several orders of magnitude larger than the classical counterpart. For this reason, it is likely that many old problems of classical numerical analysis will become relevant again in the context of quantum computing.

### C. Parameter-shift gradient estimators

In Sec. III we derived, for a large class of variational circuits, an exact formula for the gradient which is given in Eq. (9). For a finite number of shots, we can define the corresponding parameter-shift estimator

$$\hat{g}_j^{(s)} = \frac{\hat{f}(\theta_j + s) - \hat{f}(\theta_j - s)}{2\sin(s)} = g_j + \frac{\hat{\epsilon}_+ - \hat{\epsilon}_-}{2\sin(s)}, \tag{43}$$

where $\hat{\epsilon}_\pm$ is the statistical noise associated with the measurement of $\hat{f}(\theta_j \pm s)$. Different from the finite-difference estimator $\hat{g}_j^{(h)}$ presented in Eq. (33), $\hat{g}_j^{(s)}$ is unbiased because in this case there is no finite-step error since Eq. (9) is exact. Specifically, we have

$$\mathrm{Bias}\big(\hat{g}_j^{(s)}\big) = 0, \tag{44}$$

$$\mathrm{Var}\big(\hat{g}_j^{(s)}\big) = \frac{\sigma_0^2(\theta_j + s) + \sigma_0^2(\theta_j - s)}{4N\sin(s)^2} \tag{45}$$

$$\approx \frac{\sigma_0^2}{2N\sin(s)^2}. \tag{46}$$

The step from Eq. (45) to Eq. (46) is justified only if Assumption 1 is valid. The MSE of the partial-shift estimator is only due to the statistical noise and, if Assumption 1 applies, this is approximated by

$$\Delta\big(\hat{g}_j^{(s)}\big) = \mathrm{Var}\big(\hat{g}_j^{(s)}\big) \approx \frac{\sigma_0^2}{2N\sin(s)^2}. \tag{47}$$

#### 1. Parameter-shift rule with maximum shift ($s = \pi/2$)

The simple expression for the MSE [Eq. (47)] implies that, under the validity of Assumption 1, the optimal shift $s^*$ is the one which maximizes the denominator of Eq. (47), i.e., $s^* = \pi/2$. This corresponds to the parameter-shift rule already studied in the literature [10–13]. The explicit definitions of the estimator and its MSE are

$$\hat{g}_j^{(s=\pi/2)} = \frac{\hat{f}(\theta_j + \pi/2) - \hat{f}(\theta_j - \pi/2)}{2} = g_j + \frac{\hat{\epsilon}_+ - \hat{\epsilon}_-}{2}, \tag{48}$$

$$\Delta\big(\hat{g}_j^{(s=\pi/2)}\big) = \mathrm{Var}\big(\hat{g}_j^{(s=\pi/2)}\big) = \frac{\sigma_0^2(\theta_j + \pi/2) + \sigma_0^2(\theta_j - \pi/2)}{2N} \tag{49}$$

$$\simeq \frac{\sigma_0^2}{2N}. \tag{50}$$

#### 2. Scaled parameter-shift gradient estimator

A simple way of further reducing the statistical error below the value of Eq. (45) is to define a scaled parameter-shift estimator, which is the same as Eq. (43) but scaled by a multiplicative constant $\lambda \in [0, 1]$:

$$\hat{g}_j^{(\lambda,s)} = \lambda \hat{g}_j^{(s)} = \lambda g_j + \lambda \frac{\hat{\epsilon}_+ - \hat{\epsilon}_-}{2\sin(s)}. \tag{51}$$

The effect of the scaling is to reduce the variance by a factor of $\lambda^2$. However, it has a cost: The new estimator is not unbiased anymore. Indeed, we have

$$\mathrm{Bias}\big(\hat{g}_j^{(\lambda,s)}\big) = (\lambda - 1)g_j, \tag{52}$$

$$\mathrm{Var}\big(\hat{g}_j^{(\lambda,s)}\big) = \lambda^2 \mathrm{Var}\big(\hat{g}_j^{(s)}\big), \tag{53}$$

and so from Eq. (32) its MSE is

$$\Delta\big(\hat{g}_j^{(\lambda,s)}\big) = \lambda^2 \mathrm{Var}\big(\hat{g}_j^{(s)}\big) + (\lambda - 1)^2 g_j^2. \tag{54}$$

The error is quadratic with respect to $\lambda$ and is minimized by

$$\lambda^* = \frac{1}{1 + \frac{\mathrm{Var}(\hat{g}_j^{(s)})}{g_j^2}}, \tag{55}$$

corresponding to the MSE

$$\Delta\big(\hat{g}_j^{(\lambda^*,s)}\big) = \lambda^* \Delta\big(\hat{g}_j^{(\lambda=1,s)}\big) = \lambda^* \mathrm{Var}\big(\hat{g}_j^{(s)}\big) \tag{56}$$

or, equivalently, to

$$\Delta\big(\hat{g}_j^{(\lambda^*,s)}\big) = (1 - \lambda^*) g_j^2. \tag{57}$$

Equation (56) shows that the scaled estimator is always more accurate than the simple parameter-shift estimator of Eq. (48). Equation (57) is also interesting since it implies that the relative MSE of $\hat{g}_j^{(\lambda^*,s)}$ is always less than 1 for any amount of statistical noise and so for any $N$.

We envisage that this estimator could potentially be helpful when faced with the so-called barren plateau phenomenon [32], according to which the typical magnitude of the gradient of a random circuit decays exponentially with respect to the number of qubits. When the gradient is much smaller than the statistical estimation error [$\mathrm{Var}(\hat{g}_j^{(s)}) \gg g_j$], the optimal scaling factor $\lambda^*$ can be much smaller than the vanilla value of 1. So, in this kind of noise-dominated regime, the scaled estimator can be particularly advantageous if we wish to reduce the mean-square error. Note also that from a practical point of view, since $\lambda^*$ is a constant scalar, it can be simply absorbed into a renormalization of the learning rate in most machine learning optimizers. This observation could shed some light on the practical applicability of machine learning algorithms even when the gradient is so small that its estimation is dominated by statistical noise.

### D. Comparison between analytic and finite-difference gradient estimators

Even if some generalization approaches have been proposed [18,19], it is important to stress that all the parameter-shift methods can be directly applied only to a class of circuits (those involving involutory generators), while the operating regime of the finite-difference method is more general. However, in all those cases in which both approaches could be applied, which is the optimal one?

To give an answer to this question, it is important to look for a fair comparison and to choose a reasonable figure of merit. Since the parameter-shift rule involves two symmetric shifts, it is reasonable to compare it to the symmetric-difference estimator and not to the forward-difference one. As a figure of merit we chose the MSE between the estimated value and the true value of the gradient, i.e., Eqs. (28) and (29).

As a first step, we are going to make a comparison between $\hat{g}_j^{(h)}$ and $\hat{g}_j^{(s)}$. Eventually, we will take into consideration also the scaled parameter-shift estimator $\hat{g}_j^{(\lambda,s)}$ defined in Eq. (51), which will be shown to outperform the other estimators, assuming the optimal weighting parameter is known.

#### 1. *Small step limit $h \to 0$*

In the small step limit $h \to 0$, we notice that the finite-difference estimator (33) is approximately equal to the parameter-shift estimator (43) with $s = h$. Indeed, in this limit, $\sin(h) \simeq h$ and so

$$\hat{g}_j^{(h)} \xrightarrow{h\to 0} \hat{g}_j^{(s=h)}. \tag{58}$$

We have already shown that, if the noise is independent of the variational parameter, the MSE for the parameter-shift estimator is minimized when $s = \pi/2$ [see Eq. (50)]. So, in the small $h$ limit, we have

$$\Delta\big(\hat{g}_j^{(h)}\big) \xrightarrow{h\to 0} \Delta\big(\hat{g}_j^{(s=h)}\big) > \Delta\big(\hat{g}_j^{(s=\pi/2)}\big), \tag{59}$$

where the last inequality is valid only under the validity of Assumption 1. In this limit, for $s = \pi/2$ we get a large improvement of the error since we have $\Delta(\hat{g}_j^{(h)})/\Delta(\hat{g}_j^{(s=\pi/2)}) \propto 1/h^2$.

#### 2. *Finite step case with $h \leqslant 1$*

For a noninfinitesimal $h \leqslant 1$, we have a similar result. Indeed, for all $h \leqslant 1$, there always exists an $s_h$ such that $\sin(s_h) = h$. Thus

$$\begin{aligned}\Delta\big(\hat{g}_j^{(h)}\big) &= \Delta\big(\hat{g}_j^{(s=s_h)}\big) + \mathrm{Bias}\big(\hat{g}_j^{(h)}\big)^2 \\ &\geqslant \Delta\big(\hat{g}_j^{(s=s_h)}\big),\end{aligned} \tag{60}$$

where in the first equality we used Eq. (32), the fact that $\hat{g}_j^{(h)}$ and $\hat{g}_j^{(s_h)}$ have equal variance, and that $\hat{g}_j^{(s_h)}$ is unbiased.

So, also in this case, the parameter-shift rule has a smaller error with respect to the finite-difference method. Furthermore, if Assumption 1 is valid, the estimator $g_j^{(s=\pi/2)}$ becomes optimal since $\Delta(\hat{g}_j^{(s=s_h)}) > \Delta(\hat{g}_j^{(s=\pi/2)})$.

#### 3. *Finite step case with $h > 1$*

If $h > 1$, the problem is nontrivial. In this case, the variance of the finite-difference estimator is smaller than the variance of the parameter-shift estimator for all values of $s$, i.e., $\mathrm{Var}(\hat{g}_j^{(h)}) \leqslant \mathrm{Var}(\hat{g}_j^{(s)})$. On the other hand, since $h$ is large, the bias of $\hat{g}_j^{(h)}$ can be large while $\hat{g}_j^{(s)}$ is unbiased. The trade-off between the two effects determines which method minimizes the MSE. Moreover, we should also consider that that Taylor approximation which we used for the error analysis of the finite-difference estimator is likely to be broken for $h > 1$.

It is however intuitive that when the statistical noise is extremely large, the finite-difference formula in Eq. (33) for $h > 1$ has a large denominator $2h$ which is able to attenuate the statistical fluctuations more than the parameter-shift denominator (which is equal to 2). So we may ask if by simply adding a multiplicative constant $\lambda \in [0, 1]$ to the parameter-shift estimator we could always obtain a smaller error with respect to the finite-difference method for all values of $h$ and for any value of the statistical noise. This is indeed true, as we are going to show in the following section.

#### 4. *The scaled parameter-shift estimator is always optimal*

We have just compared the error scaling of the analytic and the finite-difference estimators. In this section, we instead compare both methods with respect to the scaled parameter-shift estimator defined in Eq. (51). In particular, we are going to show that $\hat{g}_j^{(\lambda^*,s)}$, where $\lambda^*$ is the optimal scaling parameter given in Eq. (55), always has a smaller MSE when compared to $\hat{g}_j^{(h)}$ or $\hat{g}_j^{(s)}$.

Indeed, from Eq. (56) and since $\lambda^* \leqslant 1$, we directly have

$$\Delta\big(\hat{g}_j^{(s)}\big) \geqslant \lambda^* \Delta\big(\hat{g}_j^{(s)}\big) = \Delta\big(\hat{g}_j^{(\lambda^*,s)}\big). \tag{61}$$

Moreover, after recognizing that $\hat{g}_j^{(h)} = \lambda_h \hat{g}_j^{(s=h)}$ with $\lambda_h = \sin(h)/h < 1$, we have

$$\Delta\big(\hat{g}_j^{(h)}\big) = \Delta\big(\lambda_h \hat{g}_j^{(s=h)}\big) = \Delta\big(\hat{g}_j^{(\lambda=\lambda_h,s)}\big) \geqslant \Delta\big(\hat{g}_j^{(\lambda^*,s)}\big). \tag{62}$$

Therefore, the scaled parameter-shift estimator $\hat{g}_j^{(\lambda^*,s)}$ is characterized by a smaller mean-square error when compared to both the finite-difference estimator $\hat{g}_j^{(h)}$ and the (nonscaled) parameter-shift estimator $\hat{g}_j^{(s)}$.

We stress that this result can be straightforwardly generalized also to high-order derivative estimators beyond the gradient, by simply replacing the constant $\lambda_h = \sin(h)/h$ which appears in Eq. (62) with $\lambda_h = [\sin(h)/h]^d$, where $d$ is the order of the derivative tensor. Moreover, we also highlight that this result is independent on the validity of Assumption 1. However, if Assumption 1 holds, we can deduce the additional fact that the scaled parameter-shift estimator with $s = \pi/2$ is always the optimal choice since it minimizes the mean-square error.

One may wonder why a scaled version of the finite-difference estimator was not considered. The reason is that it would be exactly equivalent to the scaled parameter-shift estimator up to a renormalization of the scale factor $\lambda$. The choice of scaling the parameter-shift estimator is however more natural since in this case we have that, for a large number of shots $N$, the optimal scaling $\lambda^*$ tends to the trivial limit of 1.

As a final comment in this section, we note that the forward-difference estimator was not considered in the above comparison. On the one hand, the forward-difference estimator has a bias that scales linearly with the step size $h$, as opposed to $h^2$ for the central-difference estimator considered here. On the other hand, the forward-difference estimator can be calculated with roughly half the number of shifted expectation values. The forward-difference estimator may hence be competitive for particular situations, depending on the curvature of the cost function and on the choice of shot number and step size.

## V. FIRST- AND SECOND-ORDER OPTIMIZATION

In many variational algorithms and in quantum machine learning problems, a loss function (which may depend non-linearly on one or more expectation values) is minimized with respect to the circuit parameters $\boldsymbol{\theta}$. For simplicity, in this section we assume that our goal is to minimize a single expectation value $f(\boldsymbol{\theta})$, associated with an observable $M$ as described in Eq. (1). More general scenarios can be treated in a similar way, with only some classical overhead due to the application of the chain rule for evaluating derivatives of composite functions.

### A. Gradient descent optimizer

One of the most common optimizers is gradient descent (GD). According to this algorithm, the parameters are first initialized with an initial (usually random) guess $\boldsymbol{\theta}^{(0)}$ and then a sequence of updated configurations $\boldsymbol{\theta}^{(1)}, \boldsymbol{\theta}^{(2)}, \ldots, \boldsymbol{\theta}^{(T)}$ is recursively generated, hopefully converging to a minimum of $f(\boldsymbol{\theta})$. At each iteration, the GD update rule is

$$\boldsymbol{\theta}^{(t)} = \boldsymbol{\theta}^{(t-1)} - \eta \nabla f(\boldsymbol{\theta}^{(t-1)}), \tag{63}$$

where $\eta > 0$ is a real hyperparameter usually called the learning rate. Note that this is essentially the same algorithm in both the classical and quantum settings, so the only part of the problem which is intrinsically "quantum" is the estimation of the gradient. Indeed, GD is the default optimizer used in most quantum machine learning software frameworks [22–26]. In the previous sections of this work, we have studied different methods of estimating the quantum gradient, including the parameter-shift rule [10–13].

### B. Newton optimizer

The generalization of the parameter-shift rule to higher-order derivatives that we introduced in Eq. (21), and in particular the analytic evaluation of the Hessian and the Fubini-study metric tensor that we have discussed in the first part of this work, can be directly exploited to implement several second-order optimizers. Such methods are similar to GD but in place of the first-order update rule of Eq. (63) they have more advanced iteration rules, which take into account also some information about the curvature of the objective function (or of the quantum state) with respect to the variational parameters $\boldsymbol{\theta}$.

For example, the update rule of the Newton optimizer (see, e.g., [30]) is

$$\boldsymbol{\theta}^{(t)} = \boldsymbol{\theta}^{(t-1)} - \eta [\mathsf{H} f(\boldsymbol{\theta}^{(t-1)})]^{-1} \nabla f(\boldsymbol{\theta}^{(t-1)}), \tag{64}$$

where $[\mathsf{H} f(\boldsymbol{\theta})]^{-1}$ is the inverse of the Hessian matrix, which can be estimated using the associated parameter-shift rule presented in Eq. (11). Since the Hessian is an $m \times m$ matrix, at each iteration, the number of expectation values which need to be estimated scales as $m^2$.

### C. Diagonal Newton optimizer

The computational cost of this method can be significantly reduced if one approximates the full Hessian matrix with its diagonal part such that the update rule for the $j$th element of $\boldsymbol{\theta}^{(t)}$ simplifies to

$$\boldsymbol{\theta}_j^{(t)} = \boldsymbol{\theta}_j^{(t-1)} - \eta g_{j,j}^{-1}(\boldsymbol{\theta}^{(t-1)}) g_j(\boldsymbol{\theta}^{(t-1)}), \tag{65}$$

where we used notation of the derivative tensor of Eq. (2). The computational cost of this optimizer, which was originally introduced in classical machine learning [16], has a linear scaling with respect to $m$ and so it can be a competitive alternative to GD. Moreover, the computational overhead of this method with respect to the analytic GD optimizer is negligible. Indeed, according to Eq. (13), the diagonal of the Hessian can be evaluated with the same $\pm\pi/2$ shifts which would be measured anyway to estimate for the gradient.

### D. Quantum natural gradient optimizer

A different second-order optimizer can be obtained from the concept of quantum natural gradient [28]. The corresponding update rule is very similar to Newton's method, but here the Hessian is replaced by the Fubini-study metric tensor defined in Eq. (14):

$$\boldsymbol{\theta}^{(t)} = \boldsymbol{\theta}^{(t-1)} - \eta [F(\boldsymbol{\theta}^{(t-1)})]^{-1} \nabla f(\boldsymbol{\theta}^{(t-1)}). \tag{66}$$

As we have shown in Sec. III C, all the elements of metric tensor $F_{i,j}$ can be analytically estimated with the parameter-shift rule derived in Eq. (16).

### E. Regularization of second-order methods

All the previous second-order methods share the same structure of the update rule $\boldsymbol{\theta}^{(t)} = \boldsymbol{\theta}^{(t-1)} - \eta A^{-1} \nabla f(\boldsymbol{\theta}^{(t-1)})$, where $A$ can be the Hessian, the metric tensor or some diagonal approximation. In the field of classical optimization, it is well known that, if $A$ is not positive definite, the optimizer can converge to a maximum instead of a minimum [30]. Moreover, if one or more eigenvalues are close to zero, the norm of the inverse matrix can be so large that the algorithm becomes unstable. Therefore, it is common practice to regularize $A$ in such a way to make it sufficiently positive. For example, one can replace $A$ with $A' = A + \epsilon \mathbb{1}$, for some positive parameter $\epsilon > 0$, which corresponds to shifting all the eigenvalues by $+\epsilon$. Alternatively, one can regularize each eigenvalue $\lambda_k$ of $A$ by replacing it with $\lambda_k' = \max(\lambda_k, \epsilon)$, without changing the associated eigenvector. In our experiments we used the previous method; however, from additional numerical simulations (see Sec. 5 of the Appendix) we noticed that the second method can be much more efficient since it only perturbs $A$ in the necessary subspace and keeps $A$ invariant whenever it is sufficiently positive.

Finally, we mention the approach which was recently proposed theoretically in Ref. [27], where the inverse of the maximum eigenvalue of the Hessian is used as a learning rate. In the notation of our work, this approach could be interpreted as a particular regularization method in which $A$ is replaced by $A' = \max(\lambda_k)\mathbb{1}$. It is worth emphasizing that Ref. [27] contains a comprehensive study of the spectrum of the Hessian and its application for analyzing the landscape of the cost function. This general analysis is beyond the aim of our present work, which is instead more focused on the experimental estimation of the Hessian and its usage in the Newton optimizer.

We also highlight the detailed analysis of Ref. [33] in which the error propagation associated with the matrix inversion of $A'$ (the regularized Hessian or metric tensor) is studied in the context of second-order optimization methods. Quite interestingly, according to Ref. [33], second-order optimizers have a relatively low sensitivity to the errors in the matrix elements of $A'$ and so they can still converge towards a minimum even when the Hessian or the metric tensor is statistically noisy. This fact is consistent with the experimental results reported Sec. VI.

## VI. NUMERICAL AND HARDWARE EXPERIMENTS

This section contains details of experiments run on simulators and hardware. We use the PENNYLANE software library [22] to construct a simple variational quantum circuit and access its expectation value and gradient. The circuit is evaluated using PENNYLANE's built-in simulator as well as on hardware by using integration with the IBM Quantum Experience to execute on the `ibmq_valencia` and `ibmq_burlington` five-qubit chips.

The circuit has five parameters $\boldsymbol{\theta} = (\theta_1, \theta_2, \theta_3, \theta_4, \theta_5)$, as shown in Fig. 1. It consists of a collection of single-qubit Pauli-$X$ rotations, an entangling block, and a measurement of the Pauli-$Z$ observable on the second qubit. The circuit is chosen to provide a nontrivial gradient with a low depth and the entangling block is set to match the topology of the hardware devices. We use a fixed set of parameters $\boldsymbol{\theta}$, which is detailed in the Appendix along with the resultant analytic expectation value and gradient.

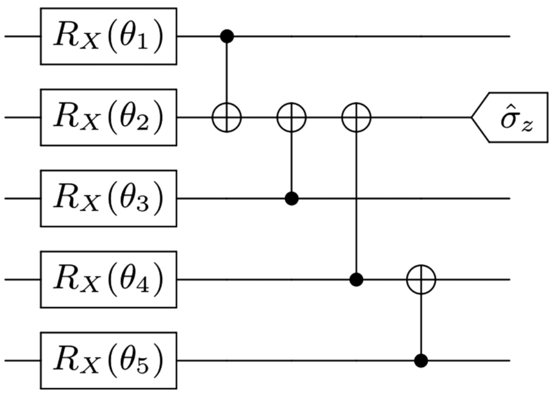


FIG. 1. Variational circuit run on a simulator and hardware to investigate the findings in this paper.

### A. Estimating the gradient

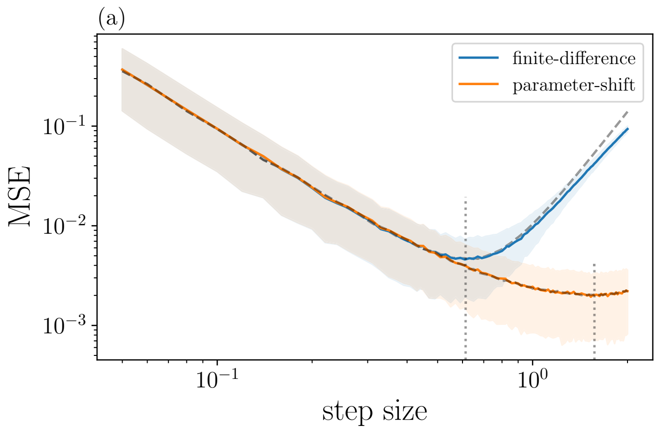
We now investigate the performance of gradient estimators in the finite-shot setting, as discussed in Sec. IV. Figure 2(a) shows how the MSE $\Delta(\hat{\boldsymbol{g}})$ scales with step size for the finite-difference and parameter-shift estimators when expectation values are estimated on a simulator with $10^3$ shots. The MSE is calculated over $10^3$ repetitions and seen to coincide closely with the theoretical predictions provided earlier. Note that the finite-difference curve starts to diverge from the predicted behavior for large step sizes due to the Taylor-series approximation breaking down in Eq. (34).

It is hence clear from the figure that the parameter-shift method with $s = \pi/2$ is the best-performing estimator in this setting. We investigate the relative performance of the two gradient estimators further using simulations in the Appendix, where we see that the parameter-shift method has the lower MSE for $N > 50$ shots. For low shot numbers, the scaled parameter-shift method mentioned in Sec. IV C 2 can also be used.

Figure 2(b) compares the gradient estimators using the `ibmq_valencia` device with $10^3$ shots per expectation value. The MSE is calculated over 16 repetitions due to limited availability of the hardware. We observe qualitative behavior similar to the simulator-based results in Fig. 2(a) but with the MSE offset upward by roughly one order of magnitude, likely due to systematic errors in the device.

### B. Estimating the Hessian

We use the results presented in Sec. III B to measure the Hessian matrix using the parameter-shift method. The Hessian of the circuit in Fig. 1 is a $(5 \times 5)$-dimensional symmetric matrix with ten off-diagonal terms and five along the diagonal. The parameter-shift method with shift $s = \pi/2$ was used to estimate the off-diagonal terms, requiring measurement of 40 expectation values. The diagonal terms were estimated with $s = \pi/4$, which required measuring 11 expectation values. The exact value of the Hessian matrix is provided in the Appendix, which also includes a simulator-based comparison of the MSE for the Hessian when estimated using the finite-difference and parameter-shift methods. The exact Hessian matrix consists of a $4 \times 4$ block of nonzero terms and a final row and column of zeros due to the expectation value being



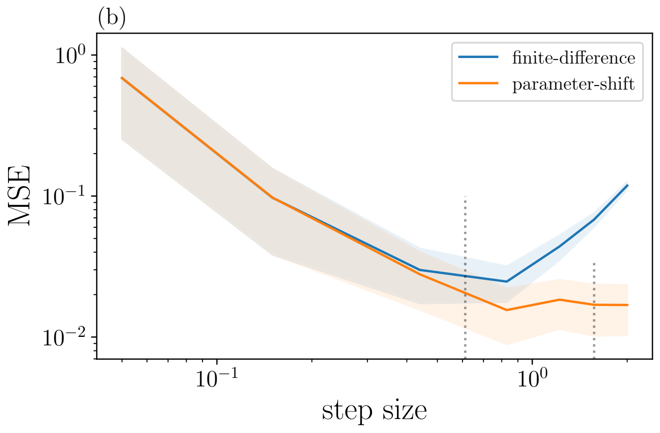


FIG. 2. Mean-square error for different choices of step size when estimating the gradient using the finite-difference and parameter-shift estimators with $10^3$ shots used to evaluate expectation values. The solid lines show the MSE, while the shaded regions illustrate the range of values within one standard deviation. Plots compare the estimators when run on (a) a simulator and (b) `ibmq_valencia`. Additional gray lines are calculated from theoretical predictions: The dashed lines show the expected behavior of the MSE and the dotted vertical lines show the expected optimal choice of step size: $h^* = 0.613$ and $s^* = \pi/2$.

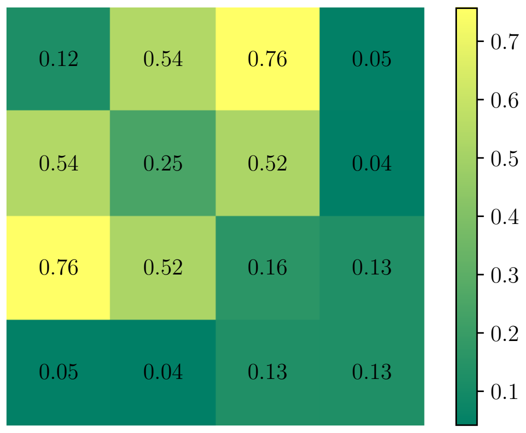


FIG. 3. Relative error for using the parameter-shift estimator to approximate the Hessian matrix on the `ibmq_valencia` hardware device. Only the upper left $4 \times 4$ block is considered, since the exact Hessian is zero in the final row and column.

independent of $\theta_5$. We estimated the Hessian matrix on hardware using `ibmq_valencia` with $N = 10^3$. Figure 3 shows the relative error for the $4 \times 4$ block, while the maximum squared error for second derivatives including the $\theta_5$ term was $8.12 \times 10^{-4}$.

From Fig. 3 one can deduce that off-diagonal terms tend to have a larger relative error. Since most of the weight of the exact Hessian is on the diagonal (see the Appendix), the error on the off-diagonal terms may be due to statistical fluctuations which have a larger impact on small matrix elements. However, given the limited amount of data, it is hard to distinguish between the errors which are caused by the experimental device and those which just correspond to statistical fluctuations.

### C. Optimization

The GD, Newton, and diagonal Newton optimizers discussed in Sec. V are now used to minimize the expectation value $f(\boldsymbol{\theta})$ of the circuit in Fig. 1 when run on hardware using the `ibmq_valencia` and `ibmq_burlington` devices. We select $\theta_1$ and $\theta_2$ as trainable parameters and leave $\theta_3$, $\theta_4$, and $\theta_5$ fixed, as well as choosing a constant learning rate of $\eta = 0.4$, as detailed in the Appendix.

A comparison of the performance of the GD optimizer when using different shot numbers $N$ to evaluate expectation values is provided in Fig. 4(a). Figure 4(a i) shows the cost function $f(\boldsymbol{\theta})$ evolving with the total number of circuit evaluations on the device, while Fig. 4(a ii) illustrates the path taken by the optimizer through the $\theta_1$-$\theta_2$ space. The starting point of the optimizer is $(\theta_1, \theta_2) = (0.1, 0.15)$ and an exact GD optimizer tends to the value $(\theta_1, \theta_2) = (0, \pi)$.

These plots highlight some interesting features of optimization on hardware. We see that the minimum can be approached even with a low shot number of $N = 10$, requiring orders of magnitude fewer circuit evaluations from the device than the $N = 100$ and $N = 1000$ cases. However, the $N = 10$ case is noisy and oscillates around the minimum point. This suggests that an adaptive shot number $N$ may be useful in practice with a low shot number initially and an increase in $N$ as the optimizer approaches a minimum, as has been discussed in recent works [34–36]. It is also important to note that the optimizer is able to approach the expected minimum even though the cost function available on the device is noisy, indicating that the direction of the gradient is still an accessible signal. Nevertheless, the gradient direction is clearly still prone to error: Note that in Fig. 4 the hardware-based paths are different from the ideal simulated path.

Figure 4(b) shows the result of using the GD, Newton, and diagonal Newton optimizers on hardware with $N = 10^3$ and with the same starting point of $(\theta_1, \theta_2) = (0.1, 0.15)$. The Newton optimizer requires evaluation of a $(2 \times 2)$-dimensional Hessian at each iteration step which is realized with nine expectation value measurements, while the diagonal Newton optimizer requires five expectation value measurements per iteration step. This contrasts with the GD optimizer which requires four evaluations per step. These differences are factored into Fig. 4(b i), which plots the cost function $f(\boldsymbol{\theta})$ in terms of the total number of circuit evaluations taken on the device.

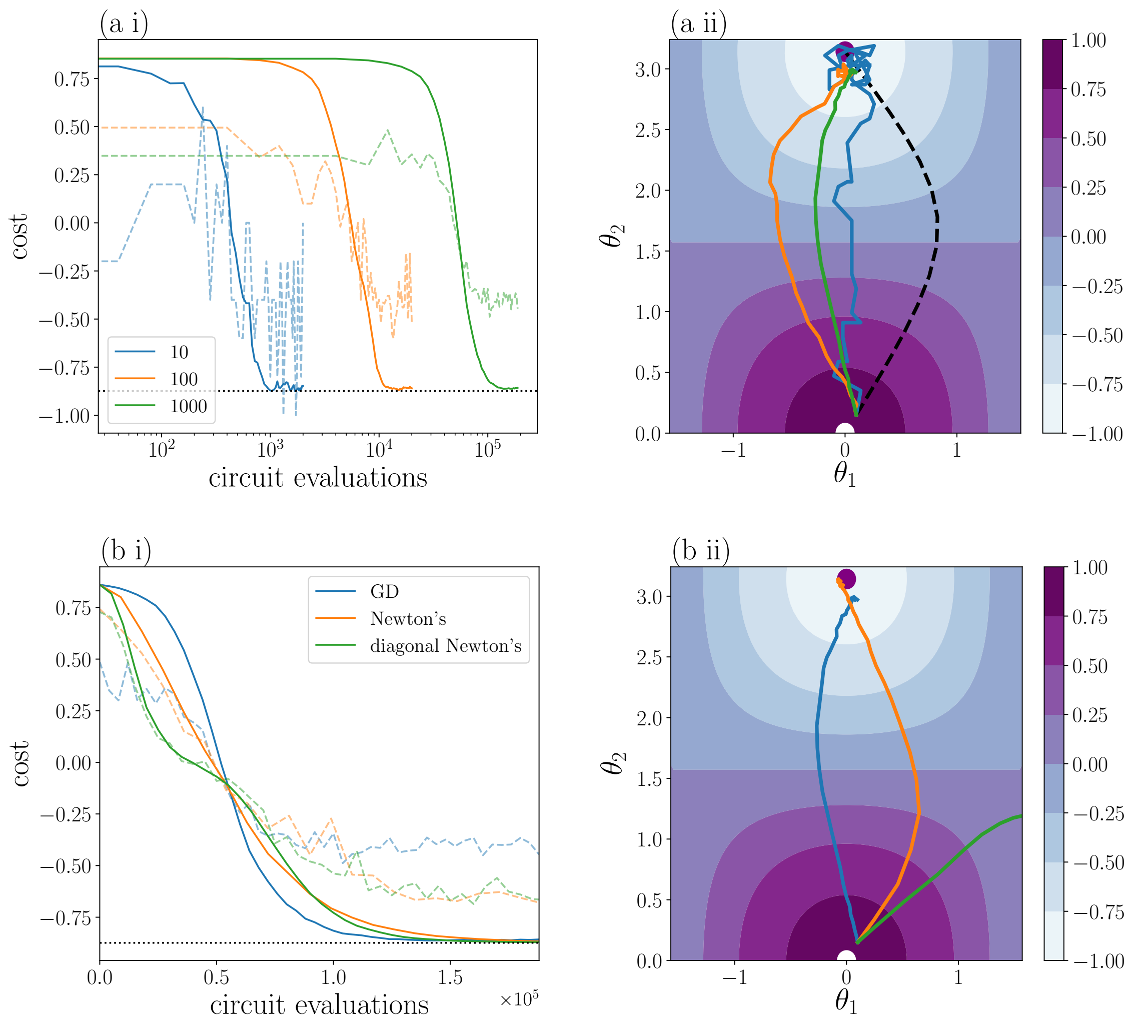


FIG. 4. Comparison of optimizers with circuit evaluation performed on the `ibmq_burlington` and `ibmq_valencia` hardware devices. The top row of plots (experiment A) focuses on the gradient descent optimizer for a varying number of shots $N$ per expectation value measurement, while the bottom row of plots (experiment B) compares different optimizers discussed in Sec. V when using $N = 1000$. (a i) and (b i) Cost function $f(\boldsymbol{\theta})$ in terms of the total number of circuit evaluations on the device and (a ii) and (b ii) the path taken by the optimizer through the $\theta_1$-$\theta_2$ space. In (a i) and (b i) the dashed lines give the cost function evaluated on hardware and the solid lines give the cost function evaluated on an exact simulator with the same $\boldsymbol{\theta}$. The dotted line gives the analytic minimum of $-0.874$. In (a ii) and (b ii) the contrasting white and purple circles highlight the maximum and minimum, while the dashed line in (a ii) is the path taken by the GD optimizer with access to exact expectation values.

We observe a comparable performance between each of the optimizers, but it is important to note that the learning rate hyperparameter and the regularization method can change the performance of each optimizer. One may also wonder why, for a large number of circuit evaluations, GD seems to be equivalent to or even better than second-order methods. The supplementary theoretical analysis reported in the Appendix suggests that this is strongly related to the choice of the regularization method (see in particular Sec. 5). The dramatic dependence of Newton optimizers on the regularization strategy is a drawback which is well known in the field of classical optimization [30]. This is a practical issue which must be taken into account also in the quantum setting.

The paths taken by each optimizer are drawn in Fig. 4(b ii). This highlights the key qualitative difference between the optimizers, which take paths based on the different information they have available, such as the local curvature information for the optimizers using the Hessian. Note that the diagonal Newton optimizer tended to the $(\pi, 0)$ minimum when evaluated on hardware. This behavior was due to the presence of noise in the device: All optimizers tend to the $(0, \pi)$ minimum with access to an ideal simulator, as shown in the Appendix.

The main scope of our experiments was to show the practical feasibility of GD in the regime of high statistical noise and to give a proof-of-principle demonstration of second-order methods on real hardware. Consistently with this scope, we used a very simple circuit, but one should be aware that, in real applications, the circuits are usually quite different (larger and more structured). A quantitative and detailed benchmarking of the different optimizers is left for future work. On this subject, we mention the recent theoretical analyses reported in Refs. [27,37].

## VII. CONCLUSION

We have derived a generalization of the parameter-shift rule for evaluating derivatives of arbitrary order. We have studied how such derivatives are affected by the statistical noise which is due to a finite number of experimental measurements, testing our predictions with numerical simulations and real quantum experiments. We have also experimentally tested how the generalized parameter-shift rules can be used to efficiently implement second-order optimization methods on near-term quantum computers.

The results presented in this work could pave the way to interesting future research directions. For example, the strong similarity between the classical *round-off* error and the quantum statistical error, which naturally emerged in our analysis of derivative estimators, is probably worth further exploration and generalization. Moreover, the analytic second-order optimizers, which in this work we tested with simple examples, could be subject to a more systematic benchmarking analysis in order to understand their competitive potential with respect to first-order methods.

## ACKNOWLEDGMENTS

We thank Josh Izaac for his help with numerical and hardware experiments. We also thank Maria Schuld for useful comments. This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Agreement No. HR00112090015 (Backpropagating Through Quantum Computers).

## APPENDIX: FURTHER DETAILS ON THE NUMERICS AND HARDWARE EXPERIMENTS

### 1. Problem setting

We use a fixed set of parameters generated from the uniform distribution over the interval $[0, 2\pi]$,

$$\boldsymbol{\theta} = (2.739, 0.163, 3.454, 2.735, 2.641). \tag{A1}$$

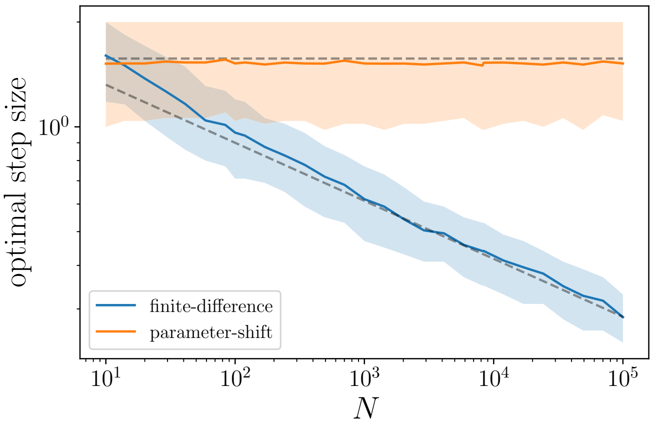


FIG. 5. Optimal step size for the finite-difference and parameter-shift gradient estimators for a range of shot numbers $N$. The shaded regions display the range of step sizes resulting in an MSE that is within 20% of the minimum observed value, while the solid lines are the medians of these regions. The dashed gray lines are the predicted values for the optimal step size.

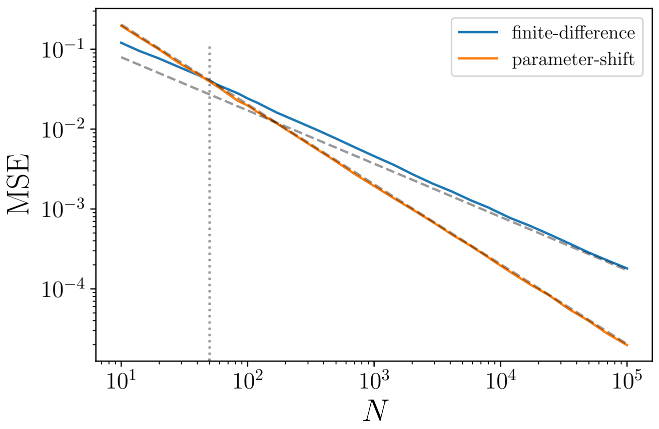


FIG. 6. Mean-square error for different choices of shot number $N$ when using the finite-difference and parameter-shift estimators with corresponding optimal choices of step size to estimate the gradient. The solid lines show the results from simulation, while the dashed gray lines are the predicted scaling. The dotted vertical line indicates the crossover point between the two methods at $N \approx 50$ shots.

Using an exact simulator, the expectation value, its gradient, and its Hessian can be evaluated as

$$f(\boldsymbol{\theta}) = -0.794, \tag{A2}$$

$$\boldsymbol{g}(\boldsymbol{\theta}) = (-0.338, 0.130, 0.256, -0.342, 0), \tag{A3}$$

$$H(\boldsymbol{\theta}) = \begin{pmatrix} 0.794 & 0.055 & 0.109 & -0.145 & 0 \\ 0.055 & 0.794 & -0.042 & 0.056 & 0 \\ 0.109 & -0.042 & 0.794 & 0.110 & 0 \\ -0.145 & 0.056 & 0.110 & 0.794 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}. \tag{A4}$$

Note that the final element of the gradient vector is exactly zero. Simulations were carried out in this paper using the `default.qubit` device in PENNYLANE with a finite

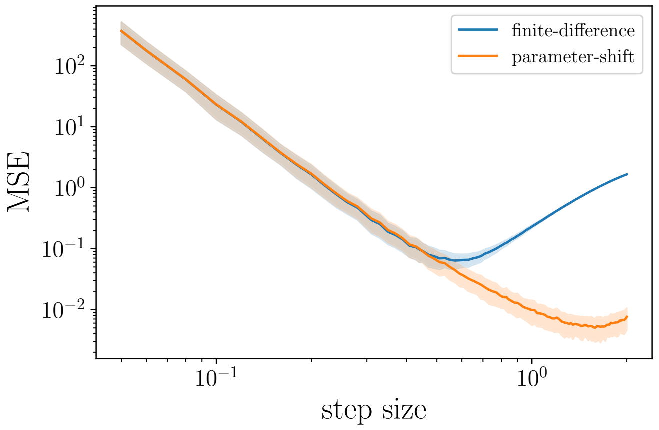


FIG. 7. Mean-square error for different choices of step size when estimating the Hessian using the finite-difference and parameter-shift estimators with $10^3$ shots used to evaluate expectation values on a simulator. The solid lines show the MSE, while the shaded regions illustrate the range of values within one standard deviation.

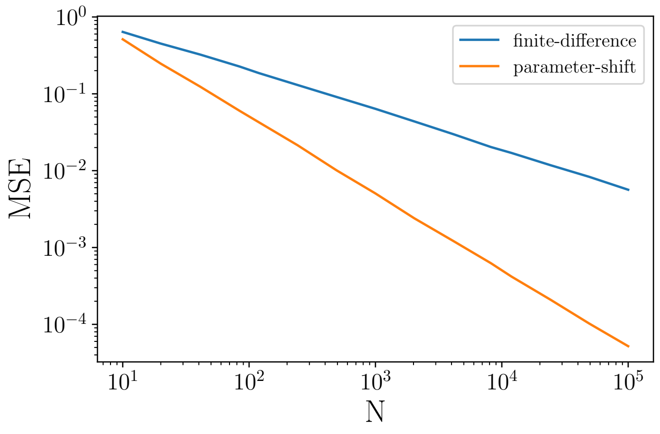


FIG. 8. Mean-square error for different choices of shot number $N$ when using the finite-difference and parameter-shift estimators with optimal choices of step size to estimate the Hessian. A power-law fit to each line gives scalings of $N^{-0.5013}$ and $N^{-1.0008}$ for the finite-difference and parameter-shift methods, respectively.

number of shots. Hardware evaluation was performed using the `qiskit.ibmq` device in the PENNYLANE-QISKIT plugin.

### 2. Estimating the gradient

Figure 5 shows the optimal step size for both the finite-difference and parameter-shift gradient estimators over a range of shot numbers $N$ used to measure expectation values on simulator. In both cases, the gradient was estimated over a fixed set of candidate choices for the step size and the one with the smallest MSE was selected. The plots also contain dashed lines illustrating the predicted value of the optimal step size. For small $N$, the optimal step size for the finite-difference estimator begins to deviate from the prediction. The reason for this deviation is that, for small $N$, the optimal step size $h^*$ is so large that the Taylor expansion around $h \simeq 0$ is not valid anymore. Figure 6 shows the MSE for the gradient estimators when using their optimal step sizes over a range of shot numbers $N$. In both Figs. 5 and 6, the MSE is calculated using 1000 repetitions.

### 3. Estimating the Hessian

Figure 7 illustrates the MSE when the Hessian is estimated using the finite-difference and parameter-shift methods for varying step sizes when expectation values are estimated on a simulator with $10^3$ shots and with $10^3$ repetitions to calculate the average. The parameter-shift estimator is calculated using Eq. (11) and the finite-difference estimator of the Hessian is taken to be [38]

$$\begin{aligned}\hat{g}^{(h)}_{j_1,j_2}(\boldsymbol{\theta}) = \big[&f\big(\boldsymbol{\theta}+h\big(\mathbf{e}_{j_1}+\mathbf{e}_{j_2}\big)\big) - f\big(\boldsymbol{\theta}+h\big(-\mathbf{e}_{j_1}+\mathbf{e}_{j_2}\big)\big)\\ &- f\big(\boldsymbol{\theta}+h\big(\mathbf{e}_{j_1}-\mathbf{e}_{j_2}\big)\big) + f\big(\boldsymbol{\theta}-h\big(\mathbf{e}_{j_1}+\mathbf{e}_{j_2}\big)\big)\big]\\ &\times (4h^2)^{-1}. \qquad \text{(A5)}\end{aligned}$$

Figure 8 compares both estimators of the Hessian for varying shot numbers $N$ when each estimator uses the corresponding optimal step size.

### 4. Second-order optimization

We minimize the expectation value $f(\boldsymbol{\theta})$ with $\boldsymbol{\theta} = (\theta_1, \theta_2, 3.454, 2.735, 2.641)$, where $\theta_1$ and $\theta_2$ are trainable parameters. The analytic minimum is equal to

$$\min_{\theta_1,\theta_2} f(\boldsymbol{\theta}) = -0.874. \qquad \text{(A6)}$$

Extremal points occur alternately when $\theta_1$ and $\theta_2$ are multiples of $\pi$.

In the results provided, `ibmq_burlington` was used when investigating the GD optimizer and `ibmq_valencia` was used when investigating the Newton and diagonal Newton optimizers. A learning rate of $\eta = 0.4$ was adopted for all optimizers. Figure 9 compares the GD, Newton, and diagonal Newton optimizers with circuit evaluation on a noise-free simulator.

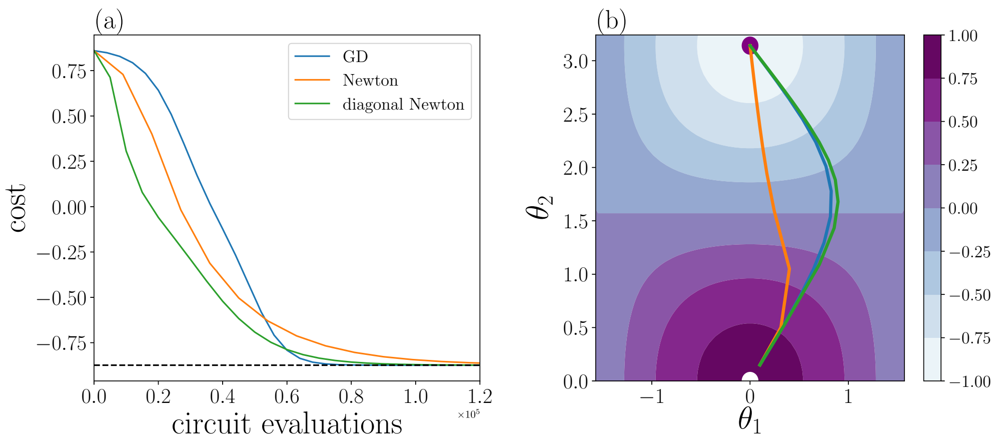


FIG. 9. Comparison of different optimizers discussed in Sec. V with circuit evaluation performed on PENNYLANE's noise-free `default.qubit` simulator. (a) Cost function $f(\boldsymbol{\theta})$ in terms of the total number of circuit evaluations and (b) path taken by the optimizer through the $\theta_1$-$\theta_2$ space. In (a) the dotted line gives the analytic minimum of $-0.874$ and in (b) the contrasting white and purple circles highlight the maximum and minimum.

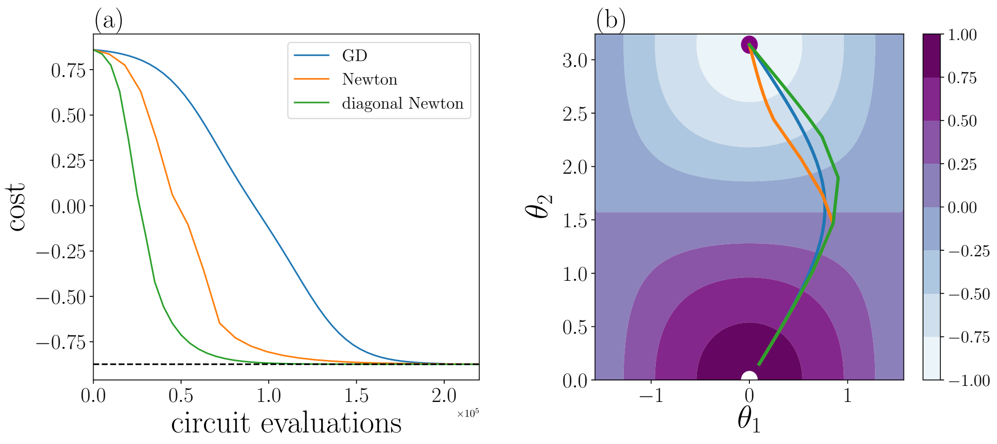


FIG. 10. Comparison of different optimizers discussed in Sec. V with circuit evaluation performed on PENNYLANE's noise-free `default.qubit` simulator. (a) Cost function $f(\boldsymbol{\theta})$ in terms of the total number of circuit evaluations and (b) path taken by the optimizer through the $\theta_1$-$\theta_2$ space. In (a) the dotted line gives the analytic minimum of $-0.874$ and in (b) the contrasting white and purple circles highlight the maximum and minimum. With respect to the simulation of Fig. 9, in this case a different regularization method is used, as discussed in the text.

### 5. Using a different regularization

As discussed in Sec. V E, one can use different regularization methods. In the previous examples we used $\mathsf{H}(f) \to \mathsf{H}(f) + \epsilon \mathbb{1}$ with $\epsilon > 0$, which in the specific cases of Figs. 9 and 4 was set to a relatively large value ($\epsilon = 1$) to compensate for the large negativity of the initial Hessian matrix. In Fig. 10, instead we numerically simulate the same optimization problem but we use a different regularization method: We replace each eigenvalue $\lambda_k$ of $\mathsf{H}(f)$ with $\max(\lambda_k, \epsilon)$, without changing the corresponding eigenvectors. In this case the results demonstrate a clear advantage of second-order optimizers with respect to gradient descent.

[1] 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).

[2] M. Schuld, A. Bocharov, K.-M. Svore, and N. Wiebe, Circuit-centric quantum classifiers, Phys. Rev. A **101**, 032308 (2020).

[3] A. Perdomo-Ortiz, M. Benedetti, J. Realpe-Gómez, and R. Biswas, Opportunities and challenges for quantum-assisted machine learning in near-term quantum computers, Quantum Sci. Technol. **3**, 030502 (2018).

[4] J.-R. McClean, J. Romero, R. Babbush, and A. Aspuru-Guzik, The theory of variational hybrid quantum-classical algorithms, New J. Phys. **18**, 023023 (2016).

[5] S. Sim, P.-D. Johnson, and A. Aspuru-Guzik, Expressibility and entangling capability of parameterized quantum circuits for hybrid quantum-classical algorithms, Adv. Quantum Technol. **2**, 1900070 (2019).

[6] 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).

[7] A. Kandala, A. Mezzacapo, K. Temme, M. Takita, M. Brink, J.-M. Chow, and J.-M. Gambetta, Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets, Nature (London) **549**, 242 (2017).

[8] E. Farhi, J. Goldstone, and S. Gutmann, A quantum approximate optimization algorithm, arXiv:1411.4028.

[9] M. Benedetti, E. Lloyd, S. Sack, and M. Fiorentini, Parameterized quantum circuits as machine learning models, Quantum Sci. Technol. **4**, 043001 (2019).

[10] J. Li, X. Yang, X. Peng, and C.-P. Sun, Hybrid Quantum-Classical Approach to Quantum Optimal Control, Phys. Rev. Lett. **118**, 150503 (2017).

[11] K. Mitarai, M. Negoro, M. Kitagawa, and K. Fujii, Quantum circuit learning, Phys. Rev. A **98**, 032309 (2018).

[12] M. Schuld, V. Bergholm, C. Gogolin, J. Izaac, and N. Killoran, Evaluating analytic gradients on quantum hardware, Phys. Rev. A **99**, 032331 (2019).

[13] K. Mitarai and K. Fujii, Methodology for replacing indirect measurements with direct measurements, Phys. RevResearch **1**, 013006 (2019).

[14] R. Cheng, Quantum geometric tensor (Fubini-Study metric) in simple quantum system: A pedagogical introduction, arXiv:1012.1337.

[15] J. Liu, H. Yuan, X.-M. Lu, and X. Wang, Quantum Fisher information matrix and multiparameter estimation, J. Phys. A: Math. Theor. **53**, 023001 (2019).

[16] S. Becker and Y. le Cun, in *Proceedings of the 1988 Connectionist Models Summer School*, edited by D. Touretzky, G. E. Hinton, and T. Sejnowski (Morgan Kaufman, San Mateo, 1988), pp. 29–37.

[17] K. Mitarai, Y.-O. Nakagawa, and W. Mizukami, Theory of analytical energy derivatives for the variational quantum eigensolver, Phys. Rev. Research **2**, 013129 (2020).

[18] G.-E. Crooks, Gradients of parameterized quantum gates using the parameter-shift rule and gate decomposition, arXiv:1905.13311.

[19] L. Banchi and G.-E. Crooks, Measuring analytic gradients of general quantum evolution with the stochastic parameter shift rule, arXiv:2005.10299.

[20] M. Ostaszewski, E. Grant, and M. Benedetti, Quantum circuit structure learning, arXiv:1905.09692.

[21] https://github.com/XanaduAI/derivatives-of-variational-circuits

[22] V. Bergholm, J. Izaac, M. Schuld, C. Gogolin, and N. Killoran, PennyLane: Automatic differentiation of hybrid quantum-classical computations, arXiv:1811.04968.

[23] M. Broughton, G. Verdon, T. McCourt, A.-J. Martinez, J. H. Yoo, S.-V. Isakov, P. Massey, M. Y. Niu, R. Halavati, E. Peters, *et al.*, TensorFlow Quantum: A software framework for quantum machine learning, arXiv:2003.02989.

[24] QULACS, https://github.com/qulacs/qulacs, 2018.

[25] TEQUILA, https://github.com/aspuru-guzik-group/tequila, 2020.

[26] X.-Z. Luo, J.-G. Liu, P. Zhang, and L. Wang, Yao.jl: Extensible, efficient framework for quantum algorithm design, Quantum **4**, 341 (2020).

[27] P. Huembeli and A. Dauphin, Characterizing the loss landscape of variational quantum circuits, arXiv:2008.02785.

[28] J. Stokes, J. Izaac, N. Killoran, and G. Carleo, Quantum natural gradient, Quantum **4**, 269 (2020).

[29] S. McArdle, T. Jones, S. Endo, Y. Li, S.-C. Benjamin, and X. Yuan, Variational ansatz-based quantum simulation of imaginary time evolution, npj Quantum Inf. **5**, 75 (2019).

[30] P.-E. Gill, W. Murray, and M.-H. Wright, *Practical Optimization* (SIAM, Philadelphia, 2019).

[31] R. Mathur, An analytical approach to computing step sizes for finite-difference derivatives, Ph.D. thesis, The University of Texas at Austin, 2012.

[32] 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).

[33] B. van Straaten and B. Koczor, Measurement cost of metric-aware variational quantum algorithms, arXiv:2005.05172.

[34] R. Sweke, F. Wilde, J. Meyer, M. Schuld, P.-K. Fährmann, B. Meynard-Piganeau, and J. Eisert, Stochastic gradient descent for hybrid quantum-classical optimization, Quantum **4**, 314 (2020).

[35] A. Arrasmith, L. Cincio, R.-D. Somma, and P.-J. Coles, Operator sampling for shot-frugal optimization in variational algorithms, arXiv:2004.06252.

[36] J.-M. Kübler, A. Arrasmith, L. Cincio, and P.-J. Coles, An adaptive optimizer for measurement-frugal variational algorithms, Quantum **4**, 263 (2020).

[37] D. Wierichs, C. Gogolin, and M. Kastoryano, Avoiding local minima in variational quantum eigensolvers with the natural gradient optimizer, Phys. Rev. Research **2**, 043246 (2020).

[38] M. Abramowitz and I. A. Stegun, *Handbook of Mathematical Functions*, 10th ed. (U.S. GPO, Washington, DC, 1972).

*Correction:* Equations (14), (16), (17), and (18) have been rescaled by a constant factor equal to $\pm 1/2$ and text above Eq. (15) has been modified.