

# A semi-agnostic ansatz with variable structure for quantum machine learning

M. Bilkis,<sup>1,2</sup> M. Cerezo,<sup>2,3</sup> Guillaume Verdon,<sup>4,5</sup> Patrick J. Coles,<sup>2</sup> and Lukasz Cincio<sup>2</sup>

<sup>1</sup>*Física Teórica: Informació i Fenòmens Quàntics, Departament de Física,  
Universitat Autònoma de Barcelona, ES-08193 Bellaterra (Barcelona), Spain*

<sup>2</sup>*Theoretical Division, Los Alamos National Laboratory, Los Alamos, NM 87545, USA*

<sup>3</sup>*Center for Nonlinear Studies, Los Alamos National Laboratory, Los Alamos, NM, USA*

<sup>4</sup>*Sandbox@Alphabet, Mountain View, CA, USA*

<sup>5</sup>*Institute for Quantum Computing, University of Waterloo, ON, Canada*

Quantum machine learning (QML) offers a powerful, flexible paradigm for programming near-term quantum computers, with applications in chemistry, metrology, materials science, data science, and mathematics. Here, one trains an ansatz, in the form of a parameterized quantum circuit, to accomplish a task of interest. However, challenges have recently emerged suggesting that deep ansatzes are difficult to train, due to flat training landscapes caused by randomness or by hardware noise. This motivates our work, where we present a variable structure approach to build ansatzes for QML. Our approach, called VAns (Variable Ansatz), applies a set of rules to both grow and (crucially) remove quantum gates in an informed manner during the optimization. Consequently, VAns is ideally suited to mitigate trainability and noise-related issues by keeping the ansatz shallow. We employ VAns in the variational quantum eigensolver for condensed matter and quantum chemistry applications, in the quantum autoencoder for data compression and in unitary compilation problems showing successful results in all cases.

Quantum computing holds the promise of providing solutions to many classically intractable problems. The availability of Noisy Intermediate-Scale Quantum (NISQ) devices [1] has raised the question of whether these devices will themselves deliver on such a promise, or whether they will simply be a stepping stone to fault-tolerant architectures.

Parameterized quantum circuits have emerged as one of the best hopes to make use of NISQ devices. Variational quantum algorithms [2–4] train such circuits to minimize a cost function and consequently accomplish a task of interest. Examples of such tasks are finding ground-states [2], solving linear systems of equations [5–7], simulating dynamics [8–11], factoring [12], compiling [13, 14], enhancing quantum metrology [15, 16], and analyzing principle components [17, 18]. More generally, in quantum machine learning (QML), one may employ multiple input states to train the parameterized quantum circuit, and many data science applications have been envisioned for QML [19–22]. We will henceforth use the term QML to encompass all of the aforementioned methods that train parameterized quantum circuits.

Despite recent relatively large-scale QML implementations [23–25], there are still several issues that need to be addressed to ensure that QML can provide a quantum advantage on NISQ devices. One issue is trainability. For instance, it has been shown that several QML architectures become untrainable for large problem sizes due to the existence of the so-called barren plateau phenomenon [26–30], which can be linked to circuits having large expressibility [31] or generating large quantities of entanglement [32–34]. The exponential scaling caused by such barren plateaus cannot simply be escaped by changing the optimizer [35, 36]. However, some promising strategies have been proposed to mitigate barren plateaus, such as correlating parameters [37], layerwise

training [38], and clever parameter initialization [39, 40].

The other major issue is quantum hardware noise, which accumulates with the circuit depth [41, 42]. This of course reduces the accuracy of observable estimation, e.g., when trying to estimate a ground state energy. However, it also leads to a more subtle and detrimental issue known as noise-induced barren plateaus [41]. Here, the noise corrupts the states in the quantum circuit and the cost function exponentially concentrates around its mean value. Similar to other barren plateaus, this phenomenon leads to an exponentially large precision being required to train the parameters. Currently, no strategies have been proposed to deal with noise-induced barren plateaus. Hence developing such strategies is a crucial research direction.

Circuit depth is clearly a key parameter for both of these issues. It is, therefore, essential to construct ansatzes that maintain a shallow depth to mitigate noise and trainability issues, but also that have enough expressibility to contain the problem solution. Two different strategies for ansatzes can be distinguished: either using a fixed [43–47] or a variable structure [48–56]. While the former is the traditional approach, the latter has recently gained considerable attention due to its versatility to address the aforementioned challenges. In variable structure ansatzes, the overall strategy consists of employing a machine learning protocol to iteratively grow the quantum circuit by placing gates that empirically lower the cost function. While these approaches address the expressibility issue by exploring specific regions of the ansatz architecture hyperspace, their depth can still grow and lead to noise-induced issues, and they can still have trainability issues from accumulating a large number of trainable parameters.

In this work, we combine several features of recently proposed methods to introduce the Variable Ansatz

(VAns) algorithm to generate variable structure ansatzes for generic QML applications. As shown in Fig. 1, VAns iteratively grows the parameterized quantum circuit by adding blocks of gates initialized to the identity, but also prevents the circuit from over-growing by removing gates and compressing the circuit at each iteration. In this sense, VAns produces shallow circuits that are more resilient to noise, and that have less trainable parameters to avoid trainability issues. Our approach provides a simple yet effective way to address the ansatz design problem, without resorting to resource-expensive computations.

This article is fix-structured as follows. In Sec. I we provide background on QML and barren plateaus, and we present a comprehensive literature review of recent variable ansatz design. We then turn to Sec. II, where we present the VAns algorithm. In Sec. III we present numerical results where we employ VAns to obtain the ground state of condensed matter and quantum chemistry systems. Here we also show how VAns can be used to build ansatzes for quantum autoencoding applications, a paradigmatic QML implementation. We also demonstrate how VAns can be applied to compile quantum circuits. Finally, in Sec. IV we discuss our results and present potential future research directions employing the VAns algorithm.

## I. Background

### A. Theoretical Framework

In this work we consider generic Quantum Machine Learning (QML) tasks where the goal is to solve an optimization problem encoded into a cost function of the form

$$C(\mathbf{k}, \boldsymbol{\theta}) = \sum_i f_i (\text{Tr}[O_i U(\mathbf{k}, \boldsymbol{\theta}) \rho_i U^\dagger(\mathbf{k}, \boldsymbol{\theta})]) . \quad (1)$$

Here,  $\{\rho_i\}$  are  $n$ -qubit states forming a training set, and  $U(\mathbf{k}, \boldsymbol{\theta})$  is a quantum circuit parametrized by continuous parameters  $\boldsymbol{\theta}$  (e.g., rotation angles) and by discrete parameters  $\mathbf{k}$  (e.g., gate placements). Moreover,  $O_i$  are observables and  $f_i$  are functions that encode the optimization task at hand. For instance, when employing the Variational Quantum Eigensolver (VQE) algorithm we have  $f_i(x) = x$  and the cost function reduces to  $C(\mathbf{k}, \boldsymbol{\theta}) = \text{Tr}[H U(\mathbf{k}, \boldsymbol{\theta}) \rho U^\dagger(\mathbf{k}, \boldsymbol{\theta})]$ , where  $\rho$  is the input state (and the only state in the training set) and  $H$  is the Hamiltonian whose ground state one seeks to prepare. Alternatively, in a binary classification problem where the training set is of the form  $\{\rho_i, y_i\}$ , with  $y_i \in \{0, 1\}$  being the true label, the choice  $f_i(x) = (x - y_i)^2$  leads to the least square error cost.

Given the cost function, a quantum computer is employed to estimate each term in (1), while the power of classical optimization algorithms is leveraged to solve the

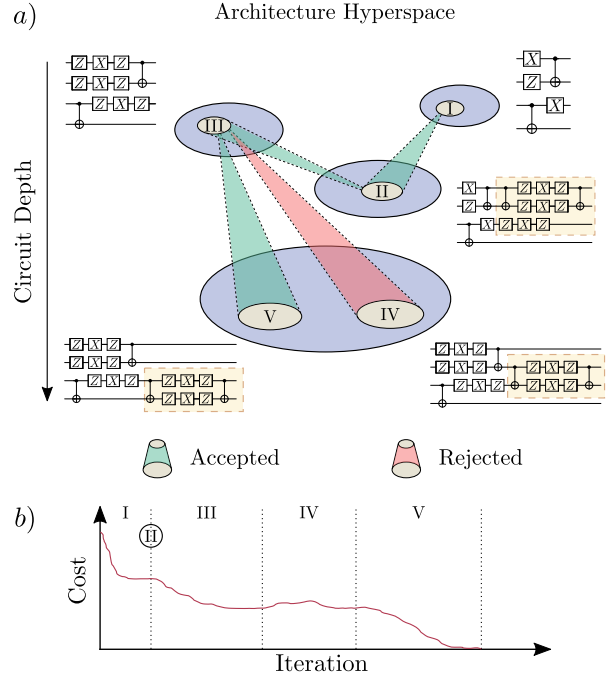


FIG. 1. **Schematic diagram of the VAns algorithm.** a) VAns explores the hyperspace of architectures of parametrized quantum circuits to create short depth ansatzes for QML applications. VAns takes a (potentially non-trivial) initial circuit (step I) and optimizes its parameters until convergence. At each step, VAns inserts blocks of gates into the circuit which are initialized to the identity (indicated in a box in the figure), so that the ansatzes at contiguous steps belong to an equivalence class of circuits leading to the same cost value (step II). VAns then employs a classical algorithm to simplify the circuit by eliminating gates and finding the shortest circuit (step II to III). The ovals represent subspaces of the architecture hyperspace connected through VAns. While some regions may be smoothly connected by placing identity resolutions, VAns can also explore regions that are not smoothly connected via a gate-simplification process. VAns can either reject (step IV) or accept (step V) modifications in the circuit structure. Here  $Z$  ( $X$ ) indicates a rotation about the  $z$  ( $x$ ) axis. b) Schematic representation of the cost function value versus the number of iterations for a typical VAns implementation which follows the steps in a).

optimization task

$$\arg \min_{\mathbf{k}, \boldsymbol{\theta}} C(\mathbf{k}, \boldsymbol{\theta}) . \quad (2)$$

The success of the QML algorithm in solving (2) hinges on several factors. First, the classical optimizer must be able to efficiently train the parameters, and in the past few years, there has been a tremendous effort in developing quantum-aware optimizers [57–63]. Moreover, while several choices of observables  $\{O_i\}$  and functions  $\{f_i\}$  can lead to different faithful cost functions (i.e., cost functions whose global optima correspond to the solution of

the problem), it has been shown that global cost functions can lead to barren plateaus and trainability issues for large problem sizes [27, 32]. Here we recall that global cost functions are defined as ones where  $O_i$  acts non-trivially on all  $n$  qubits. Finally, as discussed in the next section, the choice of ansatz for  $U(\mathbf{k}, \boldsymbol{\theta})$  also plays a crucial role in determining the success of the QML scheme.

## B. Barren Plateaus

The barren plateau phenomenon has recently received considerable attention as one of the main challenges to overcome for QML architectures to outperform their classical counterparts. Barren plateaus were first identified in [26], where it was shown that deep random parametrized quantum circuits that approximate 2-designs have gradients that are (in average) exponentially vanishing with the system size. That is, one finds that

$$\text{Var} \left[ \frac{\partial C(\mathbf{k}, \boldsymbol{\theta})}{\partial \theta} \right] \leq F(n), \quad \text{with} \quad F(n) \in \mathcal{O} \left( \frac{1}{2^n} \right), \quad (3)$$

where  $\theta \in \boldsymbol{\theta}$ . From Chebyshev's inequality we have that  $\text{Var} \left[ \frac{\partial C(\mathbf{k}, \boldsymbol{\theta})}{\partial \theta} \right]$  bounds the probability that the cost-function partial derivative deviates from its mean value (of zero) as

$$\Pr \left[ \left| \frac{\partial C(\mathbf{k}, \boldsymbol{\theta})}{\partial \theta} \right| \geq c \right] \leq \frac{\text{Var} \left[ \frac{\partial C(\mathbf{k}, \boldsymbol{\theta})}{\partial \theta} \right]}{c^2}, \quad (4)$$

for any  $c > 0$ . Hence, when the cost exhibits a barren plateau, an exponentially large precision is needed to determine a cost minimizing direction and navigate the flat landscape [35, 36].

This phenomenon was generalized in [27] to shallow circuits, and it was shown that the locality of the operators  $O_i$  in (1) play a key role in leading to barren plateaus. Barren plateaus were later analyzed and extended to the context of dissipative [32] and convolutional quantum neural networks [28, 30], and to the problem of learning scramblers [29]. A key aspect here is that circuits with large expressibility [31] (i.e., which sample large regions of the unitary group [64]) and which generate large amounts of entanglement [32–34] will generally suffer from barren plateaus. While several strategies have been developed to mitigate the randomness or entanglement in ansatzes prone to barren plateaus [18, 28, 37–40, 65, 66], it is widely accepted that designing smart ansatzes which prevent altogether barren plateaus is one of the most promising applications.

Here we remark that there exists a second method leading to barren plateaus which can even affect smart ansatzes with no randomness or entanglement-induced barren plateaus. As shown in [41], the presence of certain noise models acting throughout the circuit maps the input state toward the fixed point of the noise model

(i.e., the maximally mixed state) [41, 42], which effectively implies that the cost function value concentrates exponentially around its average as the circuit depth increases. Explicitly, in a noise-induced barren plateau one now finds that

$$\left| \frac{\partial C(\mathbf{k}, \boldsymbol{\theta})}{\partial \theta} \right| \leq g(n), \quad \text{with} \quad g(l) \in \mathcal{O} \left( \frac{1}{q^l} \right), \quad (5)$$

where  $q > 1$  is a noise parameter and  $l$  the number of layers. From Eq. (5) we see that noise-induced barren plateaus will be critical for circuits whose depth scales (at least linearly) with the number of qubits. It is worth remarking that (5) is no longer probabilistic as the whole landscape flattens. Finally, we note that strategies aimed at reducing the randomness of the circuit cannot generally prevent the cost from having a noise-induced barren plateau, since here reducing the circuit noise (improving the quantum hardware) and employing shallow circuits seem to be the only viable and promising strategies to prevent these barren plateaus.

## C. Ansatz for Parametrized Quantum Circuits

Here we analyze different ansatzes strategies for parametrized quantum circuits and how they can be affected by barren plateaus. Without loss of generality, a parametrized quantum circuit  $U(\mathbf{k}, \boldsymbol{\theta})$  can always be expressed as

$$U(\mathbf{k}, \boldsymbol{\theta}) = \prod_j U_{k_j}(\theta_j) W_{k_j}, \quad (6)$$

where  $W_{k_j}$  are fixed gates, and where  $U_{k_j}(\theta_j) = e^{-i\theta_j G_{k_j}}$  are unitaries generated by a Hermitian operator  $G_{k_j}$  and parametrized by a continuous parameter  $\theta_j \in \boldsymbol{\theta}$ . In a *fixed structure* ansatz, the discrete parameters  $k_j \in \mathbf{k}$  usually determine the type of gate, while in a *variable structure* ansatz they can also control the gate placement in the circuit.

### 1. Fixed Structure Ansatz

Let us first discuss fixed structure ansatzes. A common architecture with fixed structure is the layered Hardware Efficient Ansatz (HEA) [43], where the gates are arranged in a brick-like fashion and act on alternating pairs of qubits. One of the main advantages of this ansatz is that it employs gates native to the specific device used, hence avoiding unnecessary depth overhead arising from compiling non-native unitaries into native gates. This type of ansatz is *problem-agnostic*, in the sense that it is expressible enough so that it can be generically employed for any task. However, its high expressibility [31] can lead to trainability and barren plateau issues.

As previously mentioned, designing smart ansatzes can help in preventing barren plateaus. One such strategy

are the so-called *problem inspired* ansatzes. Here the goal is to encode information of the problem into the architecture of the ansatz so that the optimal solution of (2) exists within the parameter space without requiring high expressibility. Examples of these fixed structure ansatzes are the Unitary Coupled Cluster (UCC) Ansatz [44, 45] for quantum chemistry and the Quantum Alternating Operator Ansatz (QAOA) for optimization [46, 47]. However, while these ansatzes might not exhibit expressibility-induced barren plateaus, they usually require deep circuits to be implemented, and hence are very prone to be affected by noise-induced barren plateaus [41].

## 2. Variable Structure Ansatzes

To avoid some of the limitations of these fixed structure ansatzes, there has recently been great effort put forward towards developing variable ansatz strategies for parametrized quantum circuits [48–56]. Here, the overall strategy consists of iteratively changing the quantum circuit by placing (or removing) gates that empirically lower the cost function. In what follows we briefly review some of these variable ansatz proposals.

The first proposal for variable ansatzes for quantum chemistry was introduced in [48] under the name of ADAPT-VQE. Here, the authors follow a circuit structure similar to that used in the UCC ansatz and propose to iteratively grow the circuit by appending gates that implement fermionic operators chosen from a pool of single and double excitation operators. At each iteration, one decides which operator in the pool is to be appended, which can lead to a considerable overhead if the number of operators in the pool is large. Similarly to fix structure UCC ansatzes, the mapping from fermions to qubits can lead to prohibitively deep circuits. This issue can be overcome using the qubit-ADAPT-VQE [49] algorithm, where the pool of operators is modified in such a way that only easily implementable gates are considered. However, the size of the pool still grows with the number of qubits. In this context, trainability issues have been tackled in [67], where the parameter optimization is turned into a series of optimizations each performed on a subset of the whole parameter set; an heuristic method is proposed in order to remove gates and add new ones, which in turn allowed the authors to optimize circuits naively containing more than a thousand parameters. In addition, Ref. [50] studies how the pool of operators can be further reduced by computing the mutual information between the qubits in classically approximated ground state. We remark that estimations of the mutual information have also been recently employed to reduce the depth of fixed structured ansatzes [68]. We refer the reader to [69] for a detailed comparison between ADAPT-VQE and UCC ansatzes. Despite constituting a promising approach, it is unclear whether ADAPT-VQE and its variants will be able to overcome typical noise-

induced trainability problems as the systems under study are increased in size. Moreover, due to its specific quantum chemistry scope, the application of these schemes is limited.

A different approach to variable ansatzes that has gained considerable attention are machine-learning-aided evolutionary algorithms that upgrade individuals (quantum circuits) from a population. Noticeably, the presence of quantum correlations makes it so that it is not straightforward to combine features between circuits during the evolution, as simply merging two promising circuits does not necessarily lead to low cost function values. Thus, only random mutations have been considered so far. An example of this method is found in the Evolutionary VQE (EVQE) [51], where one smoothly explores the Hilbert space by growing the circuit with identity-initialized blocks of gates and randomly removing sequences of gates. As suggested by the authors, this might avoid entering regions leading to barren plateaus. Another example of an evolutionary algorithm is the Multi-objective Genetic VQE (MoG-VQE) [52], where one uses block-structured ansatz and simultaneously optimizes both the energy and number of entangling gates. Evolutionary algorithms constitute a promising approach to ansatzes design, they nevertheless come at the cost of high quantum-computational resources to evolve populations of quantum circuits.

A different machine learning approach to discover ansatz structures has been considered in [53, 54] where the goal is to obtain a short depth version of a given unitary. Given specific quantum hardware constraints (such as connectivity, noise-model as represented by quantum channels or available gates), an algorithm grows and modifies the structure of a parametrized quantum circuit to best match the action of the trained circuit with that of the target unitary. At each iteration, a parallelization and compression procedure is applied. This method was able to discover a short-depth version of the swap test to compute state overlaps [54], and in [53] it was shown to drastically improve the discovered circuit performance in the presence of noise. Moreover, this technique has recently been tested in large-scale numerics for combinatorial optimization problems [70]. In addition, in [71] the authors present a different iterative algorithm where single-qubit rotations are used for growing the circuit, hence leading to a scheme with limited expressibility power.

Finally, in the recent works of Refs. [55, 56, 72] the authors employ tools from auto-machine learning to build variable ansatzes. Specifically, in [55] the authors make use of the supernet and weight sharing strategies from neural network architecture search [73], while in [56] the proposal is based on a generalization of the differentiable neural architecture search [74] for quantum circuits. In [72] the authors study the design of quantum circuits for multi-label classification: policy gradient methods are used to train a neural network, so to decide which gates will compose candidate quantum cir-

cuits. We remark that while promising, the latter methods require quantum-computational resources which considerably grow with the problem sizes.

In the next section, we present a task-oriented NISQ-friendly approach to the problem of ansatzes design. In the context of the literature, the approach presented here generalizes the work in [53, 54] as a method to build potentially trainable short-depth ansatz for general quantum machine learning tasks. Unlike previous methods, the algorithm introduced here not only grows the circuit but more importantly employs classical routines to remove quantum gates in an informed manner during the optimization.

## II. The Variable Ansatz (VAns) Algorithm

### A. Overview

The goal of the VAns algorithm is to adaptively construct shallow and trainable ansatzes for generic quantum machine learning applications using parametrized quantum circuits. Let us define as  $\mathcal{C}_l$  the architecture hyperspace of quantum circuits of depth  $l$ , where a single layer is defined as gates acting in parallel. VAns takes as input:

- A cost function  $C(\mathbf{k}, \boldsymbol{\theta})$  to minimize.
- A dictionary  $\mathcal{D}$  of parametrized gates that compile to the identity. That is, for  $V(\boldsymbol{\gamma}) \in \mathcal{D}$  there exists a set of parameters  $\boldsymbol{\gamma}^*$  such that  $V(\boldsymbol{\gamma}^*) = \mathbb{I}$ .
- An initial circuit configuration  $U^{(0)}(\mathbf{k}, \boldsymbol{\theta}) \in \mathcal{C}_{l_0}$  of depth  $l_0$ .
- Circuit **Insertion** rules which stochastically take an element  $V(\boldsymbol{\gamma}^*) \in \mathcal{D}$  and append it to the circuit. The insertion is a map  $\mathcal{I} : \mathcal{C}_l \rightarrow \mathcal{C}_{l'}$  with  $l' \geq l$ .
- Circuit **Simplification** rules to eliminate unnecessary gates, redundant gates, or gates that do not have a large effect on the cost. The simplification is a map  $\mathcal{S} : \mathcal{C}_l \rightarrow \mathcal{C}_{l'}$  with  $l' \leq l$ .
- An optimization algorithm for the continuous parameters  $\boldsymbol{\theta}$ , and an optimization algorithm for the discrete parameters  $\mathbf{k}$ .

Given these inputs, VAns outputs a circuit architecture and set of parameters that approximately minimize (2). In what follows we describe the overall structure of VAns (presented in Algorithm 1), and in the next sections we provide additional details for the **Insertion** and **Simplification** modules. In all cases, the steps presented here are aimed at giving a general overview of the method and are intended to be used as building blocks for more advanced versions of VAns.

The first ingredient for VAns (besides the cost function, which is defined by the problem at hand) is a dictionary

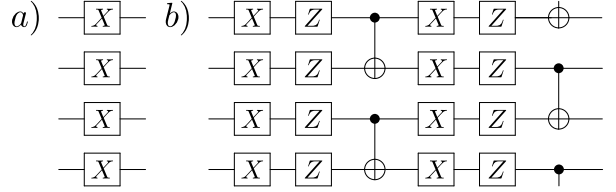


FIG. 2. **Examples of initial circuit configurations for VAns.** VAns take as input an initial structure for the parametrized quantum circuit. In (a) we depict a separable product ansatz which generates no entanglement between the qubits. On the other hand, (b) shows two layers of a shallow alternating Hardware Efficient Ansatz where neighboring qubits are initially entangled. Here  $Z$  ( $X$ ) indicates a rotation about the  $z$  ( $x$ ) axis.

$\mathcal{D}$  of parametrized gates that can compile to identity and which VAns will employ to build the ansatz. A key aspect here is that  $\mathcal{D}$  can be composed of any set of gates, so that one can build a dictionary specifically tailored for a given application. For instance, for problems with a given symmetry,  $\mathcal{D}$  can contain gates preserving said symmetry [75]. In addition, it is usually convenient to have the unitaries in  $\mathcal{D}$  expressed in terms of gates native to the specific quantum hardware employed, as this avoids compilation depth overheads.

Once the gate dictionary is set, the ansatz is initialized to a given configuration  $U^{(0)}(\mathbf{k}, \boldsymbol{\theta})$ . As shown in Algorithm 1, one then employs an optimizer to train the continuous parameters  $\boldsymbol{\theta}$  in the initial ansatz until the optimization algorithm converges. In Fig. 2, we show two non-trivial initialization strategies employed in our numerical simulations (see Section III). In Fig. 2(a) the circuit is initialized to a separable product ansatz which generates no entanglement, while in Fig. 2(b) one initializes to a shallow alternating Hardware Efficient Ansatz such that neighboring qubits are entangled. While the choice of an appropriate initial ansatz can lead to faster convergence, VAns can in principle transform a simple initial ansatz into a more complex one as part of its architecture search.

From this point, VAns enters a nested optimization loop. In the outer loop, VAns explores the architecture hyperspace to optimize the ansatz's discrete parameters  $\mathbf{k}$  that characterize the circuit structure. Then, in the inner loop, the ansatz structure is fixed and the continuous parameters  $\boldsymbol{\theta}$  are optimized.

At the start of the outer loop, VAns employs its **Insertion** rules to stochastically grow the circuit. The fact that these rules are stochastic guarantees that different runs of VAns explore distinct regions of the architecture hyperspace. As previously mentioned, the gates added to the circuit compile to the identity so that circuits that differ by gate insertions belong to an equivalence class of circuits leading to the same cost function value. As discussed below, the **Insertion** rules can be such that they depend on the current circuit they act



upon. For instance, VAns can potentially add entangling gates to qubits that were not previously connected via two-qubit gates.

To prevent the circuit from constantly growing each time gates are inserted, VAns follows the **Insertion** step by a **Simplification** step. Here, the goal is to determine if the circuit depth can be reduced without significantly modifying the cost function value in a systematic way, as proposed in [76]. This is a fundamental step of VAns as it allows the algorithm to explore and jump between different regions of the architecture hyperspace which might not be trivially connected. Moreover, unlike other variable ansatz strategies which continuously increase the circuit depth or which randomly remove gates, the **Simplification** step allows VAns to find short depth ansatzes by deleting gates in an informed manner.

Taken together, **Insertion** and **Simplification** provide a set of discrete parameters  $\mathbf{k}$ . However, to determine if this new circuit structure can improve the cost function value it is necessary to enter the inner optimization loop and train the continuous parameters  $\theta$ . When convergence in the optimization is reached, the final cost function value is compared to the cost in the previous iteration. Updates that lead to equi-cost values or to smaller costs are accepted, while updates leading to higher cost functions are accepted with exponentially decaying probability in a manner similar to a Metropolis-Hastings step [77]. Here one accepts an update which increases the cost value with a probability given by  $\exp(-\beta \frac{\Delta C}{C_0})$ , with  $\frac{\Delta C}{C_0}$  being increment in the cost function with respect to the initial value, and  $\beta > 0$  a “temperature” factor. The previous optimizations in the inner and outer loop are repeated until a termination condition is reached.

## B. Insertion method

As previously mentioned, the **Insertion** step stochastically grows the circuit by inserting into the circuit a parametrized block of gates from the dictionary  $\mathcal{D}$  which compiles to the identity. It is worth noting that in practice one can allow some deviation from the identity to reach a larger gate dictionary  $\mathcal{D}$ . In turn, this permits VAns to explore regions of the architecture hyperspace that could otherwise take several iterations to be reached. In Fig. 3 we show examples of two parametrized quantum circuits that can compile to the identity.

There are many choices for how VAns determines which gates are chosen from  $\mathcal{D}$  at each iteration, and where they should be placed. When selecting gates, we have here taken a uniform sampling approach, where every sequence of gates in  $\mathcal{D}$  has an equal probability to be selected. While one could follow a similar approach for determining where said gates should be inserted, this can lead to deeper circuits with regions containing an uneven number of CNOTs. In our heuristics, VAns has a higher probability to place two-qubit gates acting on

---

## Algorithm 1: Pseudo-code for VAns

---

**Input:** Cost function  $C(\mathbf{k}, \theta)$ ; initial circuit  $U^{(0)}(\mathbf{k}, \theta)$ ; dictionary of gates  $\mathcal{D}$ ; **Insertion** rules which take gates from  $\mathcal{D}$  and appends them to a circuit; **Simplification** rules; optimization algorithm  $\text{Opt}_C$  for continuous parameters  $\theta$ ; optimization algorithm  $\text{Opt}_D$  for discrete parameters which accepts or rejects an ansatz update given changes in the cost function value; termination condition **Term**.

**Output:** Optimized ansatz  $U^{(f)}$ .

**Init:** Randomly initialize the parameters  $\theta$ ; initialize the ansatz  $U^{(f)} \leftarrow U^{(0)}(\mathbf{k}, \theta)$ ;  $C^{(f)} \leftarrow 0$ ;  $\mathbf{k}^{(f)} \leftarrow \mathbf{k}$ ;  $\theta^{(f)} \leftarrow \theta$ ;  $\mathcal{U}(\mathbf{k}, \theta) \leftarrow \mathbf{1}$ .

- 1 Optimize  $\theta$  with  $\text{Opt}_C$  and store result in  $\theta^{(f)}$ ;  $C^{(f)} \leftarrow C(\mathbf{k}, \theta)$ .
- 2 **while** *Term is false* **do**
- 3   **Accept**  $\leftarrow$  *false*
- 4   **while** *Accept is false* **do**
- 5     Use **Insertion** in  $U^{(f)}$  and store new sets of discrete parameters, continuous parameters and ansatz in  $\theta$ ,  $\mathbf{k}$ , and  $\mathcal{U}(\mathbf{k}, \theta)$ , respectively.
- 6     Use **Simplification** on  $\mathcal{U}(\mathbf{k}, \theta)$  and store new sets of discrete parameters, continuous parameters and ansatz in  $\theta$ ,  $\mathbf{k}$ , and  $\mathcal{U}(\mathbf{k}, \theta)$ , respectively.
- 7     Optimize continuous parameters in  $\mathcal{U}(\mathbf{k}, \theta)$  with  $\text{Opt}_C$  and store result in  $\theta$ ;  $C^{(f)} \leftarrow C(\mathbf{k}, \theta)$ .
- 8     Repeat step 6.
- 9     Given  $C(\mathbf{k}, \theta)$  and  $C^{(f)}$ , optimize discrete parameters in  $\mathcal{U}(\mathbf{k}, \theta)$  with  $\text{Opt}_D$  and store result in **Accept**.
- 10    $\mathbf{k}^{(f)} \leftarrow \mathbf{k}$ .
- 11    $\theta^{(f)} \leftarrow \theta$ .
- 12    $U^{(f)} \leftarrow \mathcal{U}(\mathbf{k}, \theta)$ .
- 13    $C^{(f)} \leftarrow C(\mathbf{k}, \theta)$ .

---

qubits that were otherwise not previously connected or shared a small number of entangling gates.

## C. Simplification method

The **Simplification** steps in VAns are aimed at eliminating unnecessary gates, redundant gates, or gates that do not have a large effect on the cost. For this purpose, **Simplification** moves gates in the circuit using the commutation rules shown in Fig. 4(a) to group single qubits rotations and CNOTs together. Once there are no further commutations possible, the circuit is scanned and a sequence of simplification rules are consequently applied. For instance, assuming that the input state is initialized to  $|0\rangle^{\otimes n}$ , we can define the following set of simplification rules.

1. CNOT gates acting at the beginning of the circuit are removed.

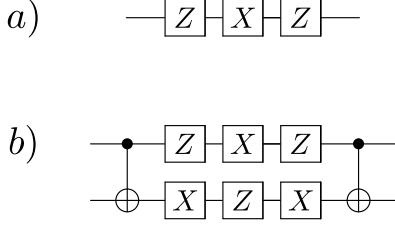


FIG. 3. **Circuits from the dictionary  $\mathcal{D}$  used during the Insertion steps.** Here we show two types of the parametrized gate sequences composed of CNOTs and rotations about the  $z$  and  $x$  axis. Specifically, one obtains the identity if the rotation angles are set to zero. Using the circuit in (a), one inserts a general unitary acting on a given qubit, while the circuit in (b) entangles the two qubits it acts upon.

2. Rotations around the  $z$ -axis acting at the beginning of the circuit are removed.
3. Consecutive CNOT sharing the same control and target qubits are removed.
4. Two or more consecutive rotations around the same axis and acting on the same qubit are compiled into a single rotation (whose value is the sum of the previous values).
5. If three or more single-qubit rotations are sequentially acting on the same qubit, they are simplified into a general single-qubit rotation of the form  $R_z(\theta_1)R_x(\theta_2)R_z(\theta_3)$  or  $R_x(\theta_1)R_z(\theta_2)R_x(\theta_3)$  which has the same action as the previous product of rotations.
6. Gates whose presence in the circuit does not considerably reduce the energy are removed.

Rules (1) – (5) are schematically shown in Fig. 4(b). We remark that a crucial feature of these **Simplification** rules is that they can be performed using a classical computer that analyzes the circuit structure and hence do not lead to additional quantum-computation resources.

As indicated by step (6), the **Simplification** steps can also delete gates whose presence in the circuit does not considerably reduce the energy. Here, given a parametrized gate, one can remove it from the circuit and compute the ensuing cost function value. If the resulting cost is increased by more than some threshold value, the gate under consideration is removed and the simplification rules (1) – (5) are again implemented. Here, one can use information from the inner optimization loop to find candidate gates for removal. For instance, when employing a gradient descent optimizer, one may attempt to remove gates whose parameters lead to small gradients. Note that, unlike the simplification steps (1) – (5) in Fig. 4(b), when using the deletion process in (6) one needs to call a quantum computer to estimate the cost function, and hence these come at an additional

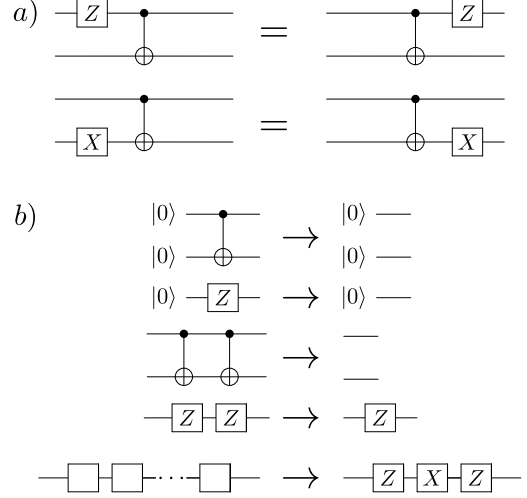


FIG. 4. **Rules for the Simplification steps.** (a) Commutation rules used by VAns to move gates in the circuit. As shown, one can commute a CNOT with a rotation  $Z$  ( $X$ ) about the  $z$  ( $x$ ) axis acting on the control (target) qubit. (b) Example of simplification rules used by VAns to reduce the circuit depth. Here we assume that the circuit is initialized to  $|0\rangle^{\otimes n}$ .

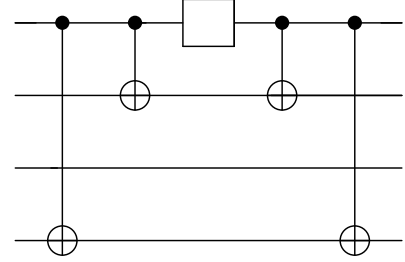


FIG. 5. **Circuit obtained from VAns.** Shown is a non-trivial circuit structure that can be obtained by VAns using the **Insertion** and **Simplification** steps and the gate dictionary in Fig. 3.

quantum-resource overhead which scales linearly with the number of gates one is attempting to remove.

An interesting aspect of the **Simplification** method is that it allows VAns to obtain circuit structures that are not contained in the initial circuit  $U^{(0)}(\mathbf{k}, \boldsymbol{\theta})$  or in the gate dictionary  $\mathcal{D}$ , and hence to explore new regions of the architecture hyperspace. For instance, using the gate dictionary in Fig. 3, VAns can obtain a gate structure as the one shown in Fig. 5.

#### D. Mitigating the effect of barren plateaus

Let us now discuss why VAns is expected to mitigate the impact of barren plateaus.

First, consider the type of barren plateaus that are

caused by the circuit approaching an approximate 2-design [26, 31]. Approximating a 2-design requires a circuit that both has a significant number of parameters and also has a significant depth. Hence, reducing either the number of parameters or the circuit depth can combat these barren plateaus. Fortunately, VAns attempts to reduce both the number of parameters and the depth. Consequently, VAns actively attempts to avoid approximating a 2-design.

Second, consider the barren plateaus that are caused by hardware noise [41]. For such barren plateaus, it was shown that the circuit depth is the key parameter, as the gradient vanishes exponentially with the depth. As VAns actively attempts to reduce the number of CNOTs, it also reduces the circuit depth. Hence VAns will mitigate the effect of noise-induced barren plateaus by keeping the depth shallow during the optimization.

### III. Numerical Results

In this section, we present heuristic results obtained from simulating VAns to solve paradigmatic problems in condensed matter, quantum chemistry, and quantum machine learning. We first use VAns in the Variational Quantum Eigensolver (VQE) algorithm [2] to obtain the ground state of the Transverse Field Ising model (TFIM), the XXZ Heisenberg spin model, and the  $H_2$  and  $H_4$  molecules. We then apply VAns to a quantum autoencoder [78] task for data compression. Finally, we use VAns to compile a Quantum Fourier Transform unitary in systems up to 10 qubits.

In all cases, the simulations were performed using Tensorflow quantum [79]; Adam [80] and qFactor [81] were employed to optimize the continuous parameters  $\theta$ . Moreover, to showcase the performance of VAns and its small quantum-computing resource requirements, we ran *a single instance* of the algorithm for each problem, and we here present the results obtained. VAns terminated once the maximum number of iterations was reached. Moreover, the simulations were performed in the absence of noise.

#### A. Transverse Field Ising model

Let us now consider a cyclic TFIM chain. The Hamiltonian of the system reads

$$H = -J \sum_{j=1}^n \sigma_j^x \sigma_{j+1}^x - g \sum_{j=1}^n \sigma_j^z, \quad (7)$$

where  $\sigma_j^{x(z)}$  is the Pauli  $x$  ( $z$ ) operator acting on qubit  $j$ , and where  $n+1 \equiv 1$  to indicate periodic boundary conditions. Here,  $J$  indicates the interaction strength, while  $g$  is the magnitude of the transverse magnetic field. As mentioned in Section I, when using the VQE algorithm the goal is to optimize a parametrized quantum circuit

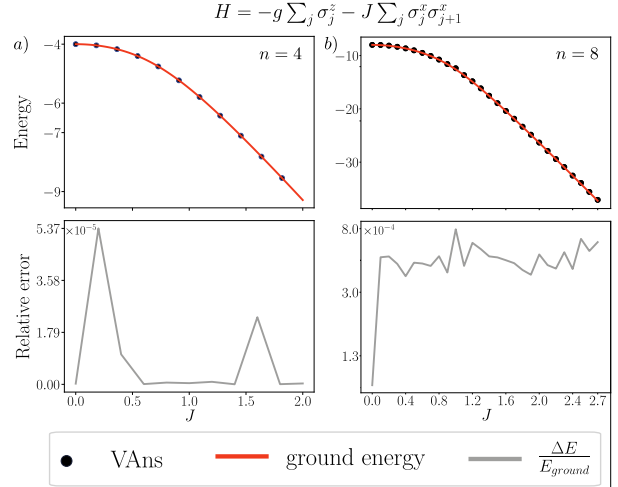


FIG. 6. Results of using VAns to obtain the ground state of a Transverse Field Ising model. Here we use VAns in the VQE algorithm for the Hamiltonian in (7) with (a)  $n = 4$  qubits and (b)  $n = 8$  qubits, field  $g = 1$ , for different values of the interaction  $J$ . Top panels: solid lines indicate the exact ground state energy, and the markers are the energies obtained using VAns. Bottom panels: Relative error in the energy for the same interaction values.

$U(\mathbf{k}, \theta)$  to prepare the ground state of  $H$  so that the cost function becomes  $E(\mathbf{k}, \theta) = \text{Tr}[HU(\mathbf{k}, \theta)\rho U^\dagger(\mathbf{k}, \theta)]$ , where one usually employs  $\rho = |\mathbf{0}\rangle\langle\mathbf{0}|$  with  $|\mathbf{0}\rangle = |0\rangle^{\otimes n}$ . Note that we here employ  $E$  as the cost function label to keep with usual notation convention.

In Fig. 6 we show results obtained from employing VAns to find the ground state of a TFIM model of Eq. (7) with  $n = 4$  qubits (a) and with  $n = 8$  qubits (b), field  $g = 1$ , and different interactions values. To quantify the performance of the algorithm, we additionally show the relative error  $|\Delta E/E_0|$ , where  $E_0$  is the exact ground state energy  $E_0$ ,  $\Delta E = E_{\text{VAns}} - E_0$ , and  $E_{\text{VAns}}$  the best energy obtained through VAns. For 4 qubits, we see from Fig. 6 that the relative error is always smaller than  $6 \times 10^{-5}$ , showing that the ground state was obtained for all cost values. Then, for  $n = 8$  qubits, VAns obtains the ground state of the TFIM with relative error smaller than  $8 \times 10^{-4}$ .

To gain some insight into the learning process, in Fig. 7 we show the cost function value, number of CNOTs, and the number of trainable parameters in the circuit discovered by VAns as different modifications of the ansatz are accepted to minimize the cost in an  $n = 8$  TFIM VQE implementation. Specifically, in Fig. 7(top) we see that as VAns explores the architecture hyperspace, the cost function value continually decreases until one can determine the ground state of the TFIM. Fig. 7(bottom) shows that initially VAns increases the number of trainable parameters and CNOTs in the circuit via the Insertion step. However, as the circuit size increases, the action



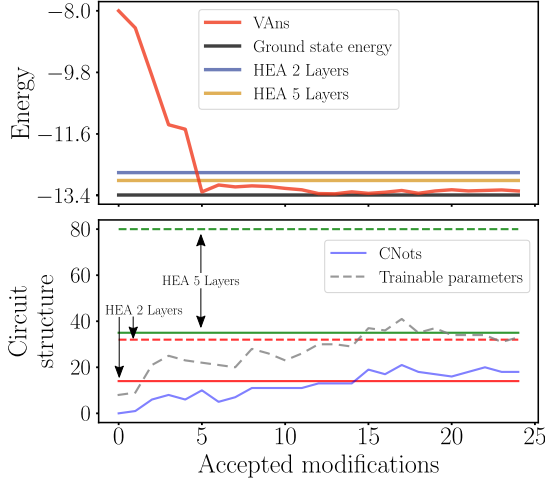


FIG. 7. **VAns learning process.** Here we show an instance of running the algorithm for the Hamiltonian in Eq. (7) with  $n = 8$  qubits, field  $g = 1$ , and interaction  $J = 1.5$ . The top panel shows the cost function value and the bottom panel depicts the number of CNOTs, and the number of trainable parameters versus the number of modifications of the ansatz accepted in the VAns algorithm. Top: As the number of iterations increases, VAns minimizes the energy until one finds the ground state of the TFIM. Here we also show the best results obtained by training a fixed structure layered Hardware Efficient Ansatz (HEA) with 2 and 5 layers, and in both cases, VAns outperforms the HEA. Bottom: While initially the number of CNOTs and number of trainable parameters increases, the Simplification method in VAns prevents the circuit from constantly growing, and can even lead to shorter depth circuits that achieve better solutions. Here we also show the number of CNOTs (solid line) and parameters (dashed line) in the HEA ansatzes considered, and we see that VAns can obtain circuits with less entangling and trainable gates.

Simplification module becomes more relevant as we see that the number of trainable parameters and CNOTs can decrease throughout the computation. Moreover, here we additionally see that reducing the number of CNOTs and trainable parameters can lead to improvements in the cost function value. The latter indicates that VAns can indeed lead to short depth ansatzes which can efficiently solve the task at hand.

In Fig. 7 we also compare the performance of VAns with that of the layered Hardware Efficient Ansatz of Fig. 2(b) with 2 and 5 layers. We specifically compare against those two fixed structure ansatzes as the first (latter) has a number of trainable parameters (CNOTs) comparable to those obtained in the VAns circuit. In all cases, we see that VAns can produce better results than those obtained with the Hardware Efficient Ansatz.

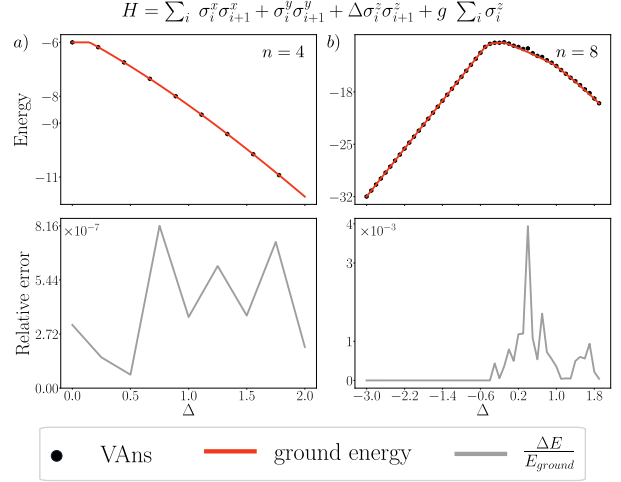


FIG. 8. **Results of using VAns to obtain the ground state of a Heisenberg  $XXZ$  model.** Here we use VAns in the VQE algorithm for the Hamiltonian in (8) with (a)  $n = 4$  and (b)  $n = 8$  qubits, field  $g = 1$ , and indicated anisotropies  $\Delta$ . Top panels: The solid line indicates the exact ground state energy, and the markers are the energies obtained using VAns. Bottom panels: Relative error in the energy for the same anisotropy values.

## B. $XXZ$ Heisenberg Model

Here we use VAns in a VQE implementation to obtain the ground of a periodic  $XXZ$  Heisenberg spin chain in a transverse field. The Hamiltonian of the system is

$$H = \sum_{j=1}^n \sigma_j^x \sigma_{j+1}^x + \sigma_j^y \sigma_{j+1}^y + \Delta \sigma_j^z \sigma_{j+1}^z + g \sum_{j=1}^n \sigma_j^z, \quad (8)$$

where again  $\sigma_j^\mu$  are the Pauli operator (with  $\mu = x, y, z$ ) acting on qubit  $j$ ,  $n+1 \equiv 1$  to indicate periodic boundary conditions, and where  $\Delta$  is the anisotropy. We recall that  $H$  commutes with the total spin component  $S_z = \sum_j \sigma_j^z$ , meaning that its eigenvectors have definite magnetization  $M_Z$  along  $z$  [82].

In Fig. 8 we show numerical results for finding the ground state of (8) with  $n = 4$  and  $n = 8$  qubits, field  $g = 1$ , and for different anisotropy values. For 4 qubits, we see that VAns can obtain the ground state with relative errors which are always smaller than  $9 \times 10^{-7}$ . In the  $n = 8$  qubits case, the relative error is of the order  $10^{-3}$ , with error increasing in the region  $0 < \Delta < 1$ . We remark that a similar phenomenon is observed in [83], where errors in preparing the ground state of the  $XXZ$  chain increase in the same region. The reason behind this phenomenon is that the optimizer can get stuck in a local minimum where it prepares excited states instead of the ground state. Moreover, it can be verified that while the ground state and the three first excited states all belong to the same magnetization sub-space of state

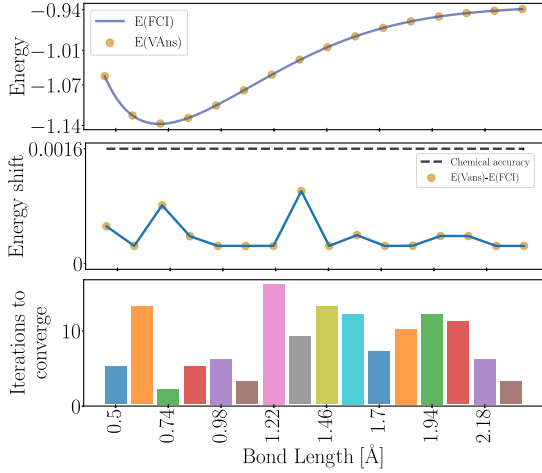


FIG. 9. **Results of using VAns to obtain the ground state of a Hydrogen molecule, at different bond lengths.** Here we use VAns in the VQE algorithm for the molecular Hamiltonian obtained after a Jordan-Wigner transformation, leading to a 4-qubit circuit. Top: Solid lines correspond to ground state energy as computed by the Full Configuration Interaction (FCI) method, whereas points correspond to energies obtained using VAns. Middle: Differences between exact and VAns ground state energies are shown. Dashed line corresponds to chemical accuracy. Bottom: Number of iterations required by VAns until convergence are shown.

with magnetization  $M_Z = 0$ , they have in fact different local symmetries and structure. Several of the low-lying excited states have a Néel-type structure of spins with non-zero local magnetization along  $z$  of the form  $|\uparrow\downarrow\uparrow\downarrow\cdots\rangle$ . On the other hand, the state that becomes the ground-state for  $\Delta > 1$  is a state where all spins have zero local magnetization along  $z$ , meaning that the local states are in the  $xy$  plane of the Bloch sphere. Since there is a larger number of excited states with a Néel-type structure (and with different translation symmetry) variational algorithms tend to prepare such states when minimizing the energy. Moreover, since mapping a state with non-zero local magnetization along  $z$  to a state with zero local magnetization requires a transformation acting on all qubits, any algorithm performing local updates will have a difficult time finding such mapping.

### C. Molecular Hamiltonians

Here we show results for using VAns to obtain the ground state of the Hydrogen molecule and the  $H_4$  chain. Molecular electronic Hamiltonians for quantum chemistry are usually obtained in the second quantization formalism in the form

$$H = \sum_{mn} h_{mn} a_m^\dagger a_n + \frac{1}{2} \sum_{mnpq} h_{mnpq} a_m^\dagger a_n^\dagger a_p a_q, \quad (9)$$

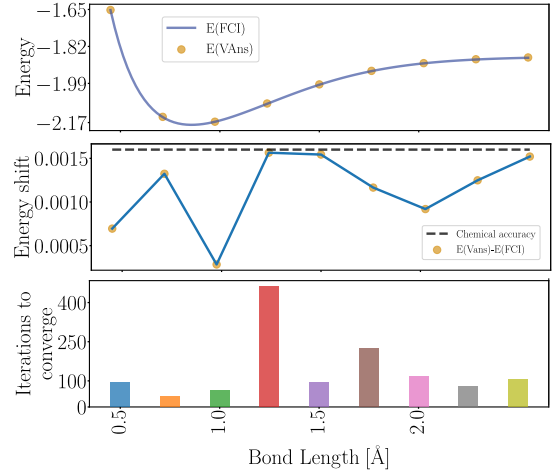


FIG. 10. **Results of using VAns to obtain the ground state of a  $H_4$  chain with a linear geometry, at different bond lengths.** Here we use VAns in the VQE algorithm for the molecular Hamiltonian obtained after a Jordan-Wigner transformation, leading to an 8-qubit circuit. Top: Solid lines correspond to ground state energy as computed by the Full Configuration Interaction (FCI) method, whereas points correspond to energies obtained using VAns. Middle: Differences between exact and VAns ground state energies are shown. Dashed line corresponds to chemical accuracy. Bottom: Number of iterations required by VAns until convergence are shown.

where  $\{a_m^\dagger\}$  and  $\{a_n\}$  are the fermionic creation and annihilation operators, respectively, and where the coefficients  $h_{mn}$  and  $h_{mnpq}$  are one- and two-electron overlap integrals, usually computed through classical simulation methods [84]. To implement (9) in a quantum computer one needs to map the fermionic operators into qubits operators (usually through a Jordan Wigner or Bravyi-Kitaev transformation). Here we employed the OpenFermion package [85] to map (9) into a Hamiltonian expressed as a linear combination of  $n$ -qubit Pauli operators of the form

$$H = \sum_{\mathbf{z}} c_{\mathbf{z}} \sigma^{\mathbf{z}}, \quad (10)$$

with  $\sigma^{\mathbf{z}} \in \{\mathbb{1}, \sigma^x, \sigma^y, \sigma^z\}^{\otimes n}$ ,  $c_{\mathbf{z}}$  real coefficients, and  $\mathbf{z} \in \{0, x, y, z\}^{\otimes n}$ .

In all cases, the basis set used to approximate atomic orbitals was STO-3g and a neutral molecule was considered. The Jordan-Wigner transformation was used in all cases. While for the  $H_2$  the number of qubits required is four ( $n = 4$ ), this number is doubled for the  $H_4$  chain ( $n = 8$ ).

#### 1. $H_2$ Molecule

Figure 9 shows the results obtained for finding the ground state of the Hydrogen molecule at different bond

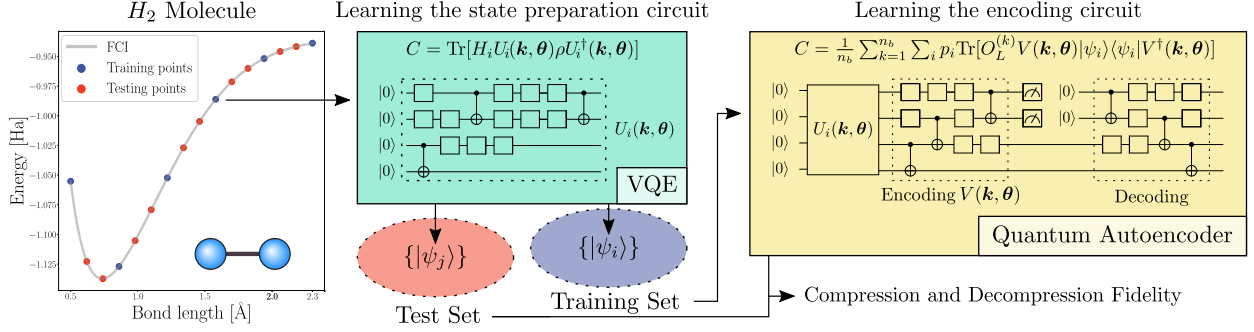


FIG. 11. **Schematic diagram of the quantum autoencoder implementation.** We first employ VAns to learn the circuits that prepare the ground states  $\{|\psi_i\rangle\}$  of the  $H_2$  molecule for different bond lengths. These ground states are then used to create a training set and test set for the quantum autoencoder implementation. The goal of the autoencoder is to train an encoding parametrized quantum circuit  $V(\mathbf{k}, \boldsymbol{\theta})$  to compress each  $|\psi_i\rangle$  into a subsystem of two qubits so that one can recover  $|\psi_i\rangle$  from the reduced states. The performance of the autoencoder can be quantified by computing the compression and decompression fidelities, i.e., the fidelity between the input state and the output state to the encoding/decoding circuit.

lengths. As shown, VAns is always able to accurately prepare the ground state within chemical accuracy. Moreover, as seen in Figure 9(bottom), VAns usually requires less than 15 iterations until achieving convergence, showing that the algorithm quickly finds a way through the architecture hyperspace towards a solution.

## 2. $H_4$ Molecule

Figure 10 shows the results obtained for finding the ground state of the  $H_4$  chain, where equal bond distances are taken. Noticeably, the dictionary of gates  $\mathcal{D}$  chosen here is not a *chemical-inspired* one (e.g., it does not contain single and double excitation operators nor its hardware-efficient implementations), yet VAns is able to find ground-state preparing circuits within chemical accuracy.

## D. Quantum Autoencoder

Here we discuss how to employ VAns and the results from the previous section for the Hydrogen molecule to train the quantum autoencoder introduced in [78]. For the sake of completeness, we now describe the quantum autoencoder algorithm for compression of quantum data.

Consider a bipartite quantum system  $AB$  of  $n_A$  and  $n_B$  qubits, respectively. Then, let  $\{p_i, |\psi_i\rangle\}$  be a training set of pure states on  $AB$ . The goal of the quantum autoencoder is to train an encoding parametrized quantum circuit  $V(\mathbf{k}, \boldsymbol{\theta})$  to compress the states in the training set onto subsystem  $A$ , so that one can discard the qubits in subsystem  $B$  without losing information. Then, using the decoding circuit (simply given by  $V^\dagger(\mathbf{k}, \boldsymbol{\theta})$ ) one can recover each state  $|\psi_i\rangle$  in the training set (or in a testing set) from the information in subsystem  $A$ . Here,  $V(\mathbf{k}, \boldsymbol{\theta})$  is essentially decoupling subsystem  $A$  from subsystem  $B$ ,

so that the state is completely compressed into subsystem  $A$  if the qubits in  $B$  are found in a fixed target state (which we here set as  $|0\rangle_B = |0\rangle^{\otimes n_B}$ ).

As shown in [78], the degree of compression can be quantified by the cost function

$$C_G(\mathbf{k}, \boldsymbol{\theta}) = 1 - \sum_i p_i \text{Tr}[(|0\rangle\langle 0|_B \otimes \mathbb{1}_A) V |\psi_i\rangle\langle\psi_i| V^\dagger],$$

where  $\mathbb{1}_B$  is the identity on subsystem  $A$ , and where we have omitted the  $\mathbf{k}$  and  $\boldsymbol{\theta}$  dependence in  $V$  for simplicity of notation. Here we see that if the reduced state in  $B$  is  $|0\rangle_B$  for all the states in the training set, then the cost is zero. Note that, as proved in [27], this is a global cost function (as one measures all the qubits in  $B$  simultaneously) and hence can have barren plateaus for large problem sizes. This issue can be avoided by considering the following local cost function where one instead measures individually each qubit in  $B$  [27]

$$C_L(\mathbf{k}, \boldsymbol{\theta}) = 1 - \sum_{k=1}^{n_B} \sum_i \frac{p_i}{n_B} \text{Tr}[(|0\rangle\langle 0|_k \otimes \mathbb{1}_{A,\bar{k}}) V |\psi_i\rangle\langle\psi_i| V^\dagger]. \quad (11)$$

Here  $\mathbb{1}_{A,\bar{k}}$  is the identity on all qubits in  $A$  and all qubits in  $B$  except for qubit  $k$ . We remark that this cost function is faithful to  $C_G(\mathbf{k}, \boldsymbol{\theta})$ , meaning that both cost functions vanish under the same conditions [27].

As shown in Fig. 11, we employ the ground states  $|\psi_i\rangle$  of the  $H_2$  molecule (for different bond lengths) to create a training set of six states and a test set of ten states. Here, the circuits obtained through VAns in the previous section serve as (fixed) state preparation circuits for the ground states of the  $H_2$  molecule. We then use VAns to learn an encoding circuit  $V(\mathbf{k}, \boldsymbol{\theta})$  which can compress the states  $|\psi_i\rangle$  into a subsystem of two qubits.

Figure 12 presents results obtained by minimizing the local cost function of (11) for a single run of the VAns algorithm. As seen, within 15 accepted architecture modifications, VAns can decrease the training cost function down to  $10^{-7}$ , by departing from a separable product

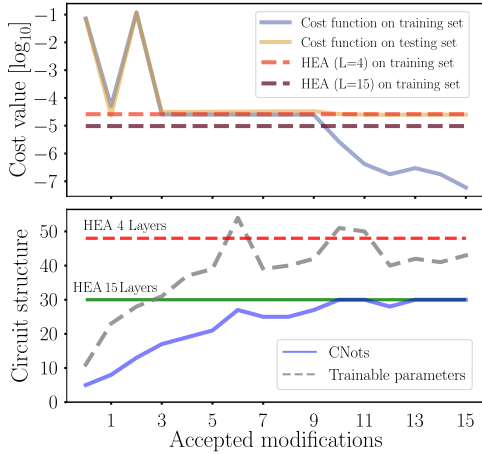


FIG. 12. **Results of using VAns to train a quantum autoencoder.** Here we use VAns to train an encoding parametrized quantum circuit by minimizing Eq. (11) on a training set comprised of six ground states of the hydrogen molecule. We here also show the lowest cost function obtained for a HEA of 4 and 15 layers. Top panel: the cost function evaluated at both versus accepted VAns circuit modifications. In addition, we also show results of evaluating the cost on the testing set. Bottom panel: number of CNOTs, and number of trainable parameters versus the number of modifications of the ansatz accepted in the VAns algorithm. Here we additionally show the number of CNOTs (solid line) and parameters (dashed line) in the HEA ansatzes considered. We remark that for  $L = 15$ , the HEA has 180 parameters, and hence the curve is not shown as it would be off the scale.

ansatz (see Fig. 2). We here additionally show results obtained by training the Hardware Efficient Ansatz of Fig. 2(b) with 4 and 15 layers (as they have a comparable number of trainable parameters and CNOTs, respectively, compared to those obtained with VAns). In all cases, VAns achieves the best performance. In particular, it is worth noting that VAns has much fewer parameters ( $\sim 45$  versus 180) than the 15-layer HEA, while also achieving a cost value that is lower by two orders of magnitude. Hence, VAns obtains better performance with fewer quantum resources.

Moreover, the fidelities obtained after the decoding process at the training set are reported in Table I. Here,  $\mathcal{F}$  is used to denote the fidelity between the input state and the state obtained after the encoding/decoding circuit. As one can see, VAns obtains very high fidelities for this task.

Set	$-\log_{10}(1 - \mathcal{F})$
Training	6.49 (5.05-6.95)
Testing	6.17 (3.61-6.95)

TABLE I. Fidelities for VAns applied to the quantum autoencoder task. The fidelities ( $\mathcal{F}$ ) are reported as Mean value (Min-Max) across both training and testing sets.

## E. Unitary compilation

Unitary compilation is a task in which a target unitary is decomposed into a sequence of quantum gates that can be implemented on a given quantum computer. Current quantum computers are limited by the depth of quantum circuits that can be executed on them, which makes the compilation task very important in the near term. Indeed, one wants to decompose a given unitary using as few gates as possible to maximally reduce the effect of noise. In this section we show that VAns is capable of finding very short decompositions as compared with other techniques.

We will illustrate our approach by compiling Quantum Fourier Transform (QFT) on systems up to  $n = 10$  qubits. Apart from VAns, we also compile the QFT unitary using standard HEA and compare the performance of both methods.

The cost function for unitary compilation is defined as follows. First, a training set is selected

$$\{(|\psi_j\rangle, U_{\text{QFT}}^{(n)}|\psi_j\rangle)\}_{j=1}^N, \quad (12)$$

where  $U_{\text{QFT}}^{(n)}$  is a target QFT unitary on  $n$  qubits and  $|\psi_j\rangle$  are  $N$ , randomly selected input states. We assume that the states  $|\psi_j\rangle$  are pairwise orthogonal to avoid potential optimization problems caused by similarities in the training set. The cost function takes the form

$$C(\mathbf{k}, \boldsymbol{\theta}) = \sum_{j=1}^N \|U_{\text{QFT}}^{(n)}|\psi_j\rangle - V(\mathbf{k}, \boldsymbol{\theta})|\psi_j\rangle\|^2. \quad (13)$$

Note that the cost function introduced in Eq. (13) becomes equivalent to a more standard one,  $C'(\mathbf{k}, \boldsymbol{\theta}) = \|U_{\text{QFT}}^{(n)} - V(\mathbf{k}, \boldsymbol{\theta})\|^2$ , when  $N = 2^n$ . While  $C(\mathbf{k}, \boldsymbol{\theta})$  measures the distance between the exact output of QFT and the one returned by  $V(\mathbf{k}, \boldsymbol{\theta})$  only on selected input states,  $C'(\mathbf{k}, \boldsymbol{\theta})$  measures the discrepancy between full unitaries  $U_{\text{QFT}}^{(n)}$  and  $V(\mathbf{k}, \boldsymbol{\theta})$ .

It has recently been shown [86] that  $N \ll 2^n$  is sufficient to accurately decompose  $U_{\text{QFT}}^{(n)}$ . More precisely, a constant number of training states  $N$  (independent of  $n$ ) can be used to ensure small value of  $C'(\mathbf{k}, \boldsymbol{\theta})$ , while minimizing the cost function in Eq. (13). This observation provides an exponential speedup in evaluating the cost function for unitary compilation. Indeed, the cost of evaluating  $C(\mathbf{k}, \boldsymbol{\theta})$  in Eq. (13) is  $N \cdot 2^n$  (assuming the circuit  $V(\mathbf{k}, \boldsymbol{\theta})$  consists of few body gates and the states  $U_{\text{QFT}}^{(n)}|\psi_j\rangle$  are given in computational basis), while the cost of computing  $C'(\mathbf{k}, \boldsymbol{\theta})$  is  $4^n$ .

The number of training states  $N$  which lead to small value of  $C'(\mathbf{k}, \boldsymbol{\theta})$  depends on the number of independent variational parameters in  $V(\mathbf{k}, \boldsymbol{\theta})$ . Suboptimal decompositions  $V(\mathbf{k}, \boldsymbol{\theta})$  (in terms of number of parametrized gates) will require larger  $N$  to achieve good compilation accuracy. We have used  $N = 15$  for  $n = 10$  qubit compilation with VAns, and a much larger training set in an

approach that uses HEA; the increase in  $N$  is necessary since the latter approach needed a much deeper circuit, as discussed below.

We have used most general two-qubit gates as the building block in VAns and to construct HEA. This is a slight generalization to the **Insertion** and **Simplification** steps in the VAns algorithm discussed above. General two-qubit gates can be decomposed down to CNOTs and one-qubit rotations using standard methods.

The method based on HEA requires very deep circuits (at  $n = 10$ ). They consist of so many gates that the regular optimization has very small success probability. We therefore modify the method based on HEA and utilize the recursive structure of  $U_{\text{QFT}}^{(n)}$ . In the modified approach, we use HEA to compile  $U_{\text{QFT}}^{(n-1)}$  and then use it to create an ansatz for  $U_{\text{QFT}}^{(n)}$ . The ansatz for larger system size additionally consists of several layers of HEA. We apply the above growth technique starting from  $n = 3$  to eventually build the ansatz for  $n = 10$ . We stress that VAns does not require such simplification and is capable of finding the decomposition with high success probability directly at  $n = 10$  while initialized randomly.

Figure 13 shows VAns results for  $n = 10$  qubit QFT compilation. Panel (a) depicts how the value of the cost function  $C(\mathbf{k}, \boldsymbol{\theta})$  is minimized over the iterations. We also show the corresponding value of  $C'(\mathbf{k}, \boldsymbol{\theta}) = \|U_{\text{QFT}}^{(n)} - V(\mathbf{k}, \boldsymbol{\theta})\|^2$ . We observe strong correlation between both cost functions.  $C'(\mathbf{k}, \boldsymbol{\theta})$  is eventually minimized below  $10^{-9}$  at the end of the optimization. Panel (b) shows how the number of two-qubit gates evolves as VAns optimization is performed. Excluding the initial “warm-up” period, during which the cost function has very large (close to maximal possible) value, the number of two-qubit gates is steadily grown reaching 48 at the end of the optimization.

The approach based on HEA requires 219 general two-qubit gates to decompose  $n = 10$  QFT, which is over 4 times more than the best circuit found by VAns. The HEA approach uses recursive structure of QFT to find accurate decomposition, while VAns does not rely on that property and finds a solution in fewer number of iterations. Finally, VAns takes advantage of the generalization bound [86], finding solution with near optimal number of variational parameters in  $V(\mathbf{k}, \boldsymbol{\theta})$ ; the training set size  $N$  required for small generalization error is small resulting in fast cost function evaluations.

#### IV. Discussion

In this work we have introduced the VAns algorithm, a semi-agnostic method for building variable structure parametrized quantum circuits. We expect VAns to be especially useful for abstract applications, such as linear systems [5–7], factoring [12], compiling [13, 14], metrology [15, 16], and data science [17–22], where physically

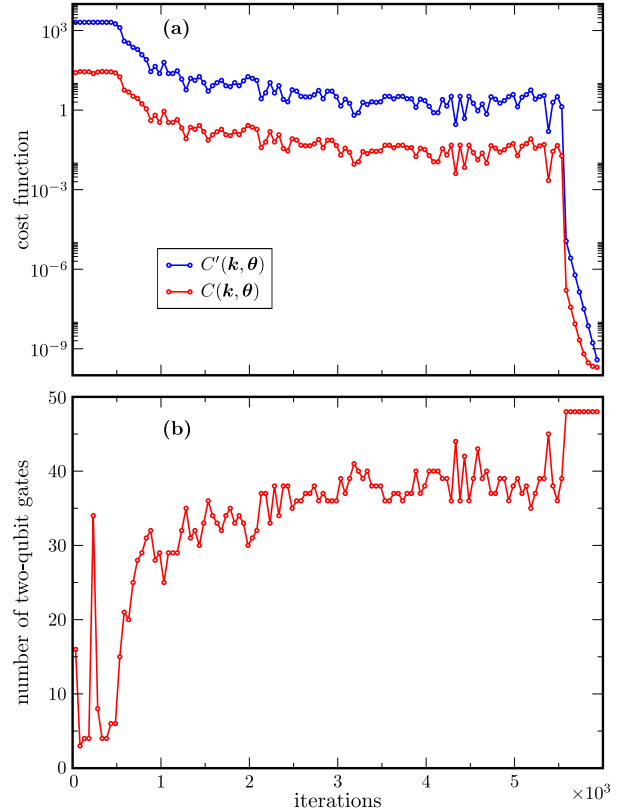


FIG. 13. **Results of using VAns for unitary compilation.** Here we use VAns to find a decomposition of QFT unitary defined on  $n = 10$  qubits, by minimizing a cost function  $C(\mathbf{k}, \boldsymbol{\theta})$  (red line in panel (a)) defined in Eq. (13). The cost evaluates a difference between exact output of QFT and the one returned by a current circuit, on a small number of input states only ( $N = 15$ ). The blue line shows corresponding difference between full unitaries,  $C'(\mathbf{k}, \boldsymbol{\theta}) = \|U_{\text{QFT}}^{(n)} - V(\mathbf{k}, \boldsymbol{\theta})\|^2$ . We observe a high correlation between those two cost functions. Panel (b) shows how VAns modifies the number of two-qubit gates as it approaches the minimum of  $C(\mathbf{k}, \boldsymbol{\theta})$ . The minimum is found with 48 gates, which is  $\sim 4.5$  times less than the decomposition found with HEA (not shown).

motivated ansatzes are not readily available. In addition, VAns will likely find use even for physical applications such as finding ground states of molecular and condensed matter systems, as it provides a shorter depth alternative to physically motivated ansatzes for mitigating the impact of noise.

At each iteration of the optimization, VAns stochastically grows the circuit to explore the architecture hyperspace. More crucially, VAns also compresses and simplifies the circuit by removing redundant gates and unimportant gates. This is a key aspect of our method, as it differentiates VAns from other variable ansatz alternatives and allows us to produce short-depth circuits, which can mitigate the effect of noise-induced barren plateaus



(NIBPs). We will further investigate this mitigation of NIBPs in future work.

To showcase the performance of VAns, we simulated our algorithm for several paradigmatic problems in quantum machine learning. Namely, we implemented VAns to find ground states of condensed matter systems and molecular Hamiltonians, for a quantum autoencoder problem and for 10-qubit QFT compilation. In all cases, VAns was able to satisfactorily create circuits that optimize the cost. Moreover, as expected, these optimal circuits contain a small number of trainable parameters and entangling gates. Here we also compared the result of VAns with results obtained using a Hardware Efficient Ansatz with either the same number of entangling gates, or the same number of parameters, and in all cases, we found that VAns could achieve the best performance.

While we provided the basic elements and structure of VAns (i.e., the gate **Insertion** and gate **Simplification** rules), these should be considered as blueprints for variable ansatzes that can be adapted and tailored to more specific applications. For instance, the gates that VAns inserts can preserve a specific symmetry in the problem. Moreover, one can cast the VAns architecture optimization (e.g., removing unimportant gates) in more advanced learning frameworks. Examples of such frameworks include supervised learning or reinforced learning schemes, which could potentially be employed to detect which gates are the best candidates for being removed.

## V. ACKNOWLEDGEMENTS

Some of the simulations presented in this paper were done using Sandbox@Alphabet’s TPU-based Floq simulator. MB is thankful to Josep Flix, John Cal-samiglia Costa and Carles Acosta for their kindness in facilitating computational resources at PIC-Barcelona. MB acknowledges support from the U.S. Department of Energy (DOE) through a quantum computing program sponsored by the Los Alamos National Laboratory (LANL) Information Science & Technology Institute, and support from the Spanish Agencia Estatal de Investigación, project PID2019-107609GB-I00, Generalitat de Catalunya CIRIT 2017-SGR-1127. MC was initially supported by the Laboratory Directed Research and Development (LDRD) program of LANL under project number 20180628ECR, and also supported by the Center for Non-linear Studies at LANL. PJC acknowledges initial support from the LANL ASC Beyond Moore’s Law project. LC was initially supported by the LDRD program of LANL under project number 20200056DR. PJC and LC were also supported by the U.S. Department of Energy (DOE), Office of Science, Office of Advanced Scientific Computing Research, under the Accelerated Research in Quantum Computing (ARQC) program.

- 
- [1] John Preskill, “Quantum computing in the nisy era and beyond,” *Quantum* **2**, 79 (2018).
  - [2] Alberto Peruzzo, Jarrod McClean, Peter Shadbolt, Man-Hong Yung, Xiao-Qi Zhou, Peter J Love, Alán Aspuru-Guzik, and Jeremy L O’Brien, “A variational eigenvalue solver on a photonic quantum processor,” *Nature communications* **5**, 1–7 (2014).
  - [3] M. Cerezo, Andrew Arrasmith, Ryan Babbush, Simon C Benjamin, Suguru Endo, Keisuke Fujii, Jarrod R McClean, Kosuke Mitarai, Xiao Yuan, Lukasz Cincio, and Patrick J. Coles, “Variational quantum algorithms,” *Nature Reviews Physics* **3**, 625–644 (2021).
  - [4] Kishor Bharti, Alba Cervera-Lierta, Thi Ha Kyaw, Tobias Haug, Sumner Alperin-Lea, Abhinav Anand, Matthias Degroote, Hermann Heimonen, Jakob S. Kottmann, Tim Menke, Wai-Keong Mok, Sukin Sim, Leong-Chuan Kwek, and Alán Aspuru-Guzik, “Noisy intermediate-scale quantum (nisq) algorithms,” *arXiv preprint arXiv:2101.08448* (2021).
  - [5] Carlos Bravo-Prieto, Ryan LaRose, M. Cerezo, Yigit Subasi, Lukasz Cincio, and Patrick Coles, “Variational quantum linear solver,” *arXiv preprint arXiv:1909.05820* (2019).
  - [6] Hsin-Yuan Huang, Kishor Bharti, and Patrick Rebentrost, “Near-term quantum algorithms for linear systems of equations,” *arXiv preprint arXiv:1909.07344* (2019).
  - [7] Xiaosi Xu, Jinzhao Sun, Suguru Endo, Ying Li, Simon C Benjamin, and Xiao Yuan, “Variational algorithms for linear algebra,” *Science Bulletin* **66**, 2181–2188 (2021).
  - [8] Xiao Yuan, Suguru Endo, Qi Zhao, Ying Li, and Simon C Benjamin, “Theory of variational quantum simulation,” *Quantum* **3**, 191 (2019).
  - [9] Cristina Cirstoiu, Zoe Holmes, Joseph Iosue, Lukasz Cincio, Patrick J Coles, and Andrew Sornborger, “Variational fast forwarding for quantum simulation beyond the coherence time,” *npj Quantum Information* **6**, 1–10 (2020).
  - [10] Benjamin Commeau, M. Cerezo, Zoë Holmes, Lukasz Cincio, Patrick J Coles, and Andrew Sornborger, “Variational hamiltonian diagonalization for dynamical quantum simulation,” *arXiv preprint arXiv:2009.02559* (2020).
  - [11] Joe Gibbs, Kaitlin Gili, Zoë Holmes, Benjamin Commeau, Andrew Arrasmith, Lukasz Cincio, Patrick J Coles, and Andrew Sornborger, “Long-time simulations with high fidelity on quantum hardware,” *arXiv preprint arXiv:2102.04313* (2021).
  - [12] Eric Anschuetz, Jonathan Olson, Alán Aspuru-Guzik, and Yudong Cao, “Variational quantum factoring,” in *International Workshop on Quantum Technology and Optimization Problems* (Springer, 2019) pp. 74–85.
  - [13] Sumeet Khatri, Ryan LaRose, Alexander Poremba, Lukasz Cincio, Andrew T Sornborger, and Patrick J Coles, “Quantum-assisted quantum compiling,” *Quantum* **3**, 140 (2019).
  - [14] Kunal Sharma, Sumeet Khatri, M. Cerezo, and Patrick J Coles, “Noise resilience of variational quantum compiling,” *New Journal of Physics* **22**, 043006 (2020).

- [15] Jacob L Beckey, M. Cerezo, Akira Sone, and Patrick J Coles, “Variational quantum algorithm for estimating the quantum fisher information,” [arXiv preprint arXiv:2010.10488](#) (2020).
- [16] Bálint Koczor, Suguru Endo, Tyson Jones, Yuichiro Matsuzaki, and Simon C Benjamin, “Variational-state quantum metrology,” *New Journal of Physics* (2020), 10.1088/1367-2630/ab965e.
- [17] Ryan LaRose, Arkin Tikku, Étude O’Neel-Judy, Lukasz Cincio, and Patrick J Coles, “Variational quantum state diagonalization,” *npj Quantum Information* **5**, 1–10 (2019).
- [18] M. Cerezo, Kunal Sharma, Andrew Arrasmith, and Patrick J Coles, “Variational quantum state eigensolver,” [arXiv preprint arXiv:2004.01372](#) (2020).
- [19] Jacob Biamonte, Peter Wittek, Nicola Pancotti, Patrick Rebentrost, Nathan Wiebe, and Seth Lloyd, “Quantum machine learning,” *Nature* **549**, 195–202 (2017).
- [20] Maria Schuld, Ilya Sinayskiy, and Francesco Petruccione, “The quest for a quantum neural network,” *Quantum Information Processing* **13**, 2567–2586 (2014).
- [21] Amira Abbas, David Sutter, Christa Zoufal, Aurélien Lucchi, Alessio Figalli, and Stefan Woerner, “The power of quantum neural networks,” *Nature Computational Science* **1**, 403–409 (2021).
- [22] Guillaume Verdon, Jacob Marks, Sasha Nanda, Stefan Leichenauer, and Jack Hidary, “Quantum hamiltonian-based models and the variational quantum thermalizer algorithm,” [arXiv preprint arXiv:1910.02071](#) (2019).
- [23] Matthew P Harrigan, Kevin J Sung, Matthew Neeley, Kevin J Satzinger, Frank Arute, Kunal Arya, Juan Atalaya, Joseph C Bardin, Rami Barends, Sergio Boixo, *et al.*, “Quantum approximate optimization of non-planar graph problems on a planar superconducting processor,” *Nature Physics* , 1–5 (2021).
- [24] Frank Arute, Kunal Arya, Ryan Babbush, Dave Bacon, Joseph C Bardin, Rami Barends, Sergio Boixo, Michael Broughton, Bob B Buckley, David A Buell, *et al.*, “Hartree-fock on a superconducting qubit quantum computer,” *Science* **369**, 1084–1089 (2020).
- [25] Pauline J Ollitrault, Abhinav Kandala, Chun-Fu Chen, Panagiotis Kl Barkoutsos, Antonio Mezzacapo, Marco Pistoia, Sarah Sheldon, Stefan Woerner, Jay M Gambetta, and Ivano Tavernelli, “Quantum equation of motion for computing molecular excitation energies on a noisy quantum processor,” *Physical Review Research* **2**, 043140 (2020).
- [26] Jarrod R McClean, Sergio Boixo, Vadim N Smelyanskiy, Ryan Babbush, and Hartmut Neven, “Barren plateaus in quantum neural network training landscapes,” *Nature communications* **9**, 1–6 (2018).
- [27] M Cerezo, Akira Sone, Tyler Volkoff, Lukasz Cincio, and Patrick J Coles, “Cost function dependent barren plateaus in shallow parametrized quantum circuits,” *Nature communications* **12**, 1–12 (2021).
- [28] Arthur Pesah, M. Cerezo, Samson Wang, Tyler Volkoff, Andrew T Sornborger, and Patrick J Coles, “Absence of barren plateaus in quantum convolutional neural networks,” *Physical Review X* **11**, 041011 (2021).
- [29] Zoë Holmes, Andrew Arrasmith, Bin Yan, Patrick J Coles, Andreas Albrecht, and Andrew T Sornborger, “Barren plateaus preclude learning scramblers,” *Physical Review Letters* **126**, 190501 (2021).
- [30] Chen Zhao and Xiao-Shan Gao, “Analyzing the barren plateau phenomenon in training quantum neural network with the zx-calculus,” [arXiv preprint arXiv:2102.01828](#) (2021).
- [31] Zoë Holmes, Kunal Sharma, M. Cerezo, and Patrick J Coles, “Connecting ansatz expressibility to gradient magnitudes and barren plateaus,” [arXiv preprint arXiv:2101.02138](#) (2021).
- [32] Kunal Sharma, M. Cerezo, Lukasz Cincio, and Patrick J Coles, “Trainability of dissipative perceptron-based quantum neural networks,” [arXiv preprint arXiv:2005.12458](#) (2020).
- [33] Taylor L Patti, Khadijeh Najafi, Xun Gao, and Susanne F Yelin, “Entanglement devised barren plateau mitigation,” *Physical Review Research* **3**, 033090 (2021).
- [34] Carlos Ortiz Marrero, Maria Kieferova, and Nathan Wiebe, “Entanglement induced barren plateaus,” [arXiv preprint arXiv:2010.15968](#) (2020).
- [35] M. Cerezo and Patrick J Coles, “Higher order derivatives of quantum neural networks with barren plateaus,” *Quantum Science and Technology* **6**, 035006 (2021).
- [36] Andrew Arrasmith, M. Cerezo, Piotr Czarnik, Lukasz Cincio, and Patrick J Coles, “Effect of barren plateaus on gradient-free optimization,” *Quantum* **5**, 558 (2021).
- [37] Tyler Volkoff and Patrick J Coles, “Large gradients via correlation in random parameterized quantum circuits,” *Quantum Science and Technology* **6**, 025008 (2021).
- [38] Andrea Skolik, Jarrod R McClean, Masoud Mohseni, Patrick van der Smagt, and Martin Leib, “Layerwise learning for quantum neural networks,” *Quantum Machine Intelligence* **3**, 1–11 (2021).
- [39] Edward Grant, Leonard Wossnig, Mateusz Ostaszewski, and Marcello Benedetti, “An initialization strategy for addressing barren plateaus in parametrized quantum circuits,” *Quantum* **3**, 214 (2019).
- [40] Guillaume Verdon, Michael Broughton, Jarrod R McClean, Kevin J Sung, Ryan Babbush, Zhang Jiang, Hartmut Neven, and Masoud Mohseni, “Learning to learn with quantum neural networks via classical neural networks,” [arXiv preprint arXiv:1907.05415](#) (2019).
- [41] Samson Wang, Enrico Fontana, Marco Cerezo, Kunal Sharma, Akira Sone, Lukasz Cincio, and Patrick J Coles, “Noise-induced barren plateaus in variational quantum algorithms,” *Nature Communications* **12**, 1–11 (2021).
- [42] Daniel Stilck Franca and Raul Garcia-Patron, “Limitations of optimization algorithms on noisy quantum devices,” [arXiv preprint arXiv:2009.05532](#) (2020).
- [43] Abhinav Kandala, Antonio Mezzacapo, Kristan Temme, Maika Takita, Markus Brink, Jerry M. Chow, and Jay M. Gambetta, “Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets,” *Nature* **549**, 242–246 (2017).
- [44] Yudong Cao, Jonathan Romero, Jonathan P Olson, Matthias Degroote, Peter D Johnson, Mária Kieferová, Ian D Kivlichan, Tim Menke, Borja Peropadre, Nicolas PD Sawaya, *et al.*, “Quantum chemistry in the age of quantum computing,” *Chemical reviews* **119**, 10856–10915 (2019).
- [45] Rodney J Bartlett and Monika Musiał, “Coupled-cluster theory in quantum chemistry,” *Reviews of Modern Physics* **79**, 291 (2007).
- [46] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann, “A quantum approximate optimization algorithm,” [arXiv preprint arXiv:1411.4028](#) (2014).

- [47] Stuart Hadfield, Zhihui Wang, Bryan O’Gorman, Eleanor G Rieffel, Davide Venturelli, and Rupak Biswas, “From the quantum approximate optimization algorithm to a quantum alternating operator ansatz,” *Algorithms* **12**, 34 (2019).
- [48] Harper R Grimsley, Sophia E Economou, Edwin Barnes, and Nicholas J Mayhall, “An adaptive variational algorithm for exact molecular simulations on a quantum computer,” *Nature communications* **10**, 1–9 (2019).
- [49] Ho Lun Tang, VO Shkolnikov, George S Barron, Harper R Grimsley, Nicholas J Mayhall, Edwin Barnes, and Sophia E Economou, “qubit-adapt-vqe: An adaptive algorithm for constructing hardware-efficient ansatzes on a quantum processor,” *arXiv preprint arXiv:1911.10205* (2019).
- [50] Zi-Jian Zhang, Thi Ha Kyaw, Jakob Kottmann, Matthias Degroote, and Alan Aspuru-Guzik, “Mutual information-assisted adaptive variational quantum eigensolver,” *Quantum Science and Technology* (2021), 10.1088/2058-9565/abdca4.
- [51] Arthur G Rattew, Shaohan Hu, Marco Pistoia, Richard Chen, and Steve Wood, “A domain-agnostic, noise-resistant, hardware-efficient evolutionary variational quantum eigensolver,” *arXiv preprint arXiv:1910.09694* (2019).
- [52] D Chivilikhin, A Samarin, V Ulyantsev, I Iorsh, AR Oganov, and O Kyriienko, “Mog-vqe: Multiobjective genetic variational quantum eigensolver,” *arXiv preprint arXiv:2007.04424* (2020).
- [53] Lukasz Cincio, Kenneth Rudinger, Mohan Sarovar, and Patrick J. Coles, “Machine learning of noise-resilient quantum circuits,” *PRX Quantum* **2**, 010324 (2021).
- [54] Lukasz Cincio, Yiğit Subaşı, Andrew T Sornborger, and Patrick J Coles, “Learning the quantum algorithm for state overlap,” *New Journal of Physics* **20**, 113022 (2018).
- [55] Yuxuan Du, Tao Huang, Shan You, Min-Hsiu Hsieh, and Dacheng Tao, “Quantum circuit architecture search: error mitigation and trainability enhancement for variational quantum solvers,” *arXiv preprint arXiv:2010.10217* (2020).
- [56] Shi-Xin Zhang, Chang-Yu Hsieh, Shengyu Zhang, and Hong Yao, “Differentiable quantum architecture search,” *arXiv preprint arXiv:2010.08561* (2020).
- [57] Guillaume Verdon, Jason Pye, and Michael Broughton, “A universal training algorithm for quantum deep learning,” *arXiv preprint arXiv:1806.09729* (2018).
- [58] Jonas M Kübler, Andrew Arrasmith, Lukasz Cincio, and Patrick J Coles, “An adaptive optimizer for measurement-frugal variational algorithms,” *Quantum* **4**, 263 (2020).
- [59] Andrew Arrasmith, Lukasz Cincio, Rolando D Somma, and Patrick J Coles, “Operator sampling for shot-frugal optimization in variational algorithms,” *arXiv preprint arXiv:2004.06252* (2020).
- [60] James Stokes, Josh Izaac, Nathan Killoran, and Giuseppe Carleo, “Quantum natural gradient,” *Quantum* **4**, 269 (2020).
- [61] Bálint Koczor and Simon C Benjamin, “Quantum natural gradient generalised to non-unitary circuits,” *arXiv preprint arXiv:1912.08660* (2019).
- [62] Ken M Nakanishi, Keisuke Fujii, and Synge Todo, “Sequential minimal optimization for quantum-classical hybrid algorithms,” *Physical Review Research* **2**, 043158 (2020).
- [63] Enrico Fontana, M. Cerezo, Andrew Arrasmith, Ivan Rungger, and Patrick J Coles, “Optimizing parametrized quantum circuits via noise-induced breaking of symmetries,” *arXiv preprint arXiv:2011.08763* (2020).
- [64] Sukin Sim, Peter D. Johnson, and Alán Aspuru-Guzik, “Expressibility and entangling capability of parameterized quantum circuits for hybrid quantum-classical algorithms,” *Advanced Quantum Technologies* **2**, 1900070 (2019).
- [65] Kaining Zhang, Min-Hsiu Hsieh, Liu Liu, and Dacheng Tao, “Toward trainability of quantum neural networks,” *arXiv preprint arXiv:2011.06258* (2020).
- [66] Kishor Bharti and Tobias Haug, “Quantum assisted simulator,” *arXiv preprint arXiv:2011.06911* (2020).
- [67] Sukin Sim, Jonathan Romero, Jérôme F Gonthier, and Alexander A Kunitsa, “Adaptive pruning-based optimization of parameterized quantum circuits,” *Quantum Science and Technology* **6**, 025019 (2021).
- [68] Nikolay V Tkachenko, James Sud, Yu Zhang, Sergei Tretiak, Petr M Anisimov, Andrew T Arrasmith, Patrick J Coles, Lukasz Cincio, and Pavel A Dub, “Correlation-informed permutation of qubits for reducing ansatz depth in vqe,” *PRX Quantum* **2**, 020337 (2021).
- [69] Daniel Claudino, Jerimiah Wright, Alexander J McCaskey, and Travis S Humble, “Benchmarking adaptive variational quantum eigensolvers,” *Frontiers in Chemistry* **8**, 1152 (2020).
- [70] Xiaoyuan Liu, Anthony Angone, Ruslan Shaydulin, Ilya Safro, Yuri Alexeev, and Lukasz Cincio, “Layer vqe: A variational approach for combinatorial optimization on noisy quantum computers,” *arXiv preprint arXiv:2102.05566* (2021).
- [71] Mateusz Ostaszewski, Edward Grant, and Marcello Benedetti, “Structure optimization for parameterized quantum circuits,” *Quantum* **5**, 391 (2021).
- [72] Mohammad Pirhooshayan and Tamas Terlaky, “Quantum circuit design search,” (2020).
- [73] Thomas Elsken, Jan Hendrik Metzen, Frank Hutter, *et al.*, “Neural architecture search: A survey,” *J. Mach. Learn. Res.* **20**, 1–21 (2019).
- [74] Hanxiao Liu, Karen Simonyan, and Yiming Yang, “Darts: Differentiable architecture search,” *arXiv preprint arXiv:1806.09055* (2018).
- [75] Bryan T Gard, Linghua Zhu, George S Barron, Nicholas J Mayhall, Sophia E Economou, and Edwin Barnes, “Efficient symmetry-preserving state preparation circuits for the variational quantum eigensolver algorithm,” *npj Quantum Information* **6**, 1–9 (2020).
- [76] D. Maslov, G. W. Dueck, D. M. Miller, and C. Negrevergne, “Quantum circuit simplification and level compaction,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* **27**, 436–444 (2008).
- [77] W. K. Hastings, “Monte carlo sampling methods using markov chains and their applications,” *Biometrika* **57**, 97–109 (1970).
- [78] Jonathan Romero, Jonathan P Olson, and Alan Aspuru-Guzik, “Quantum autoencoders for efficient compression of quantum data,” *Quantum Science and Technology* **2**, 045001 (2017).
- [79] Michael Broughton, Guillaume Verdon, Trevor McCourt, Antonio J Martinez, Jae Hyeon Yoo, Sergei V Isakov, Philip Massey, Murphy Yuezheng Niu, Ramin Halavati, Evan Peters, *et al.*, “Tensorflow quantum: A soft-

- ware framework for quantum machine learning,” *arXiv preprint arXiv:2003.02989* (2020), 10.1038/nature23474.
- [80] Diederik P Kingma and Jimmy Ba, “Adam: A method for stochastic optimization,” in *Proceedings of the 3rd International Conference on Learning Representations (ICLR)* (2015).
- [81] E. Younis and L. Cincio, “Quantum Fast Circuit Optimizer (qFactor),” .
- [82] M Cerezo, Raúl Rossignoli, N Canosa, and E Ríos, “Factorization and criticality in finite  $xxz$  systems of arbitrary spin,” *Physical Review Letters* **119**, 220605 (2017).
- [83] Alba Cervera-Lierta, Jakob S Kottmann, and Alán Aspuru-Guzik, “The meta-variational quantum eigensolver (meta-vqe): Learning energy profiles of parameterized hamiltonians for quantum simulation,” *arXiv preprint arXiv:2009.13545* (2020).
- [84] Alán Aspuru-Guzik, Anthony D. Dutoi, Peter J. Love, and Martin Head-Gordon, “Simulated quantum computation of molecular energies,” *Science* **309**, 1704–1707 (2005).
- [85] Jarrod R. McClean, Kevin J. Sung, Ian D. Kivlichan, Yudong Cao, Chengyu Dai, E. Schuyler Fried, Craig Gidney, Brendan Gimby, Pranav Gokhale, Thomas Häner, Tarini Hardikar, Vojtěch Havlíček, Oscar Higgott, Cupjin Huang, Josh Izaac, Zhang Jiang, Xinle Liu, Sam McArdle, Matthew Neeley, Thomas O’Brien, Bryan O’Gorman, Isil Ozfidan, Maxwell D. Radin, Jhonathan Romero, Nicholas Rubin, Nicolas P. D. Sawaya, Kanav Setia, Sukin Sim, Damian S. Steiger, Mark Steudtner, Qiming Sun, Wei Sun, Daochen Wang, Fang Zhang, and Ryan Babbush, “Openfermion: The electronic structure package for quantum computers,” (2019).
- [86] Matthias C. Caro, Hsin-Yuan Huang, M. Cerezo, Kunal Sharma, Andrew Sornborger, Lukasz Cincio, and Patrick J. Coles, “Generalization in quantum machine learning from few training data,” *arXiv preprint arXiv:2111.05292* (2021).