Lie-algebraic classical simulations for quantum computing

Matthew L. Goh,[1, 2, *] Martin Larocca,[1, 3] Lukasz Cincio,[1] M. Cerezo,[4] and Frédéric Sauvage[1, 5]

[1] *Theoretical Division, Los Alamos National Laboratory, Los Alamos, New Mexico 87545, USA*
[2] *Department of Materials, University of Oxford, Parks Road, Oxford OX1 3PH, United Kingdom*
[3] *Center for Nonlinear Studies, Los Alamos National Laboratory, Los Alamos, New Mexico 87545, USA*
[4] *Information Sciences, Los Alamos National Laboratory, Los Alamos, New Mexico 87545, USA*
[5] *Quantinuum, Partnership House, Carlisle Place, London SW1P 1BX, United Kingdom*

The classical simulation of quantum dynamics plays an important role in our understanding of quantum complexity, and in the development of quantum technologies. Efficient techniques such as those based on the Gottesman-Knill theorem for Clifford circuits, tensor networks for low entanglement-generating circuits, or Wick's theorem for fermionic Gaussian states, have become central tools in quantum computing. In this work, we contribute to this body of knowledge by presenting a framework for classical simulations, dubbed '$\mathfrak{g}$-sim', which is based on the underlying Lie algebraic structure of the dynamical process. When the dimension of the algebra grows at most polynomially in the system size, there exists observables for which the simulation is efficient. Indeed, we show that $\mathfrak{g}$-sim enables new regimes for classical simulations, is able to deal with certain forms of noise in the evolution, as well as can be used to tackle several paradigmatic variational and non-variational quantum computing tasks. For the former, we perform Lie-algebraic simulations to train and optimize parametrized quantum circuits (thus effectively showing that some variational models can be dequantized), design enhanced parameter initialization strategies, solve tasks of quantum circuit synthesis, and train a quantum-phase classifier. For the latter, we report large-scale noiseless and noisy simulations on benchmark problems. By comparing the limitations of $\mathfrak{g}$-sim and certain Wick's theorem-based simulations, we find that the two methods become inefficient for different types of states or observables, hinting at the existence of distinct, non-equivalent, resources for classical simulation.

I. INTRODUCTION

At a fundamental level, the ability to *classically* simulate quantum dynamics in certain regimes helps us refine our understanding of the true nature of quantum complexity. Different approaches allowing for scalable classical simulations have elucidated distinct aspects of quantumness by restricting the set of operations performed, the initial states evolved, and the observables measured. These include, but are not limited to, the role of the discrete-group structure of Clifford operations in the case of stabilizer simulations [1], the importance of entanglement in the case of tensor networks [2], and the existence of transformations to non-interacting particles in the case of free fermions [3–5]. For all the these approaches, the computational resources required for simulations scale only *polynomially* in the system size, as opposed to the *exponential* cost of the generic or unrestricted case. Such simulations are deemed classically efficient or scalable.

At a practical level, the ability to efficiently simulate quantum systems facilitates the development of quantum technology. As the size of experimentally demonstrated quantum systems is starting to exceed what can be simulated classically, it is of great importance to rely on faithful classical computational results [6], even if they are restricted to special types of quantum evolutions, input states, and observables. In this context, classical simulations can serve the role of benchmarks for currently developed devices and can be used for the purpose of characterization. Furthermore, classical simulations can find many more applications along the development and study of variational quantum computing. These encompass variational quantum algorithms (VQAs) [7, 8] and quantum machine learning (QML) methods [9–12], that most typically rely on the ability to optimize over parameters of quantum circuits. Despite the promises of such algorithms and demonstrations for relatively small system sizes, it remains unclear how viable these are when scaled to larger sizes [13–15]. As such, scalable quantum simulators can serve to study the true potential and limitations of VQAs and QML models, and to extend their capabilities. For instance, classical simulations can be used to provide exact inputs for learning-based error mitigation strategies [16–19], to probe the trainability of quantum circuits at large scale [20], for the purpose of pre-training the models [21–31], but also to dequantize certain architectures [32–42].

Among the scalable simulation methods that have been proposed, we focus on those based on the Lie-algebraic structure of the underlying dynamics [41, 43, 44]. Here, simulation complexity scales with the dimension (denoted $\dim(\mathfrak{g})$) of the Lie-algebra (denoted $\mathfrak{g}$) induced by the operators generating the dynamics of the system - these notions are defined more precisely soon. Crucially, in certain cases such dimension can be substantially smaller that the dimension of the Hilbert space of the system of interest, allowing for more efficient simulations that would be entailed generically. In particular, such simulations become scalable whenever $\dim(\mathfrak{g})$

* matt.goh@merton.ox.ac.uk

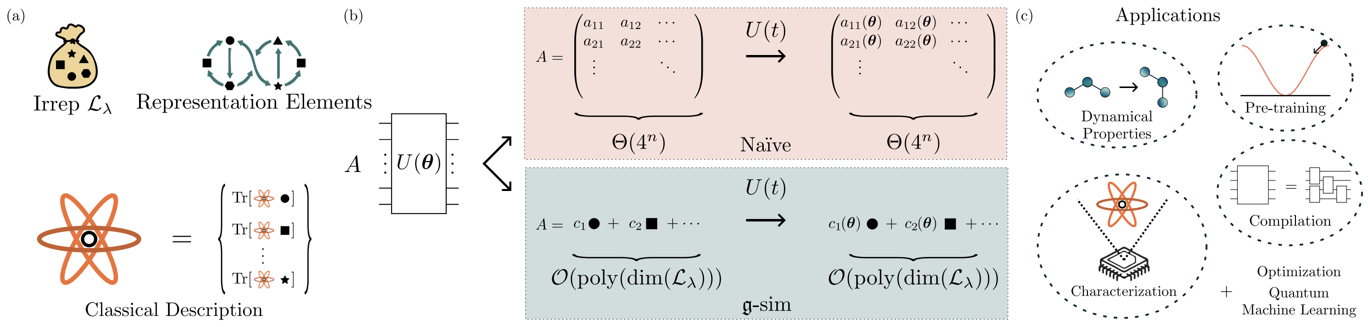


FIG. 1. **Theoretical framework and applications of 𝔤-sim.** **(a)** Essential components of the Lie-algebraic representation of quantum dynamics. First, we need a description of the Lie algebra $\mathfrak{g}$ associated with a unitary evolution of interest (3), as well as an irreducible representation (irrep) $\mathcal{L}_\lambda$: a subspace of the operator space that is invariant under the action of $\mathcal{G} = e^{\mathfrak{g}}$ (Definition. 3). Here we depict the elements of $\mathcal{L}_\lambda$ as a collection of black geometric shapes. Next, we need to know how these elements inter-connect to each other via the dynamics. This is determined entirely by the so-called representation elements (7). The final ingredient of 𝔤-sim is a classical description of the input state, i.e., the expectation values with respect to the elements of $\mathcal{L}_\lambda$ (13). **(b)** Comparison of naïve and Lie-algebraic approaches to quantum dynamics in the Heisenberg picture. In the naïve approach, an observable A is viewed as a linear operator acting on an n-qubit Hilbert space $\mathcal{H}$, with $\dim(\mathcal{H}) = 2^n$. The observable is represented as a $2^n \times 2^n$ matrix whose coefficients evolve according to the von Neumann equation of motion, and thus the approach *always* scales as $\Theta(4^n)$ and is not classically scalable. In the Lie-algebraic approach, the observable is instead viewed as a linear combination of distinct basis terms in $\mathfrak{g}$, whose coefficients couple to each other over time via the representation elements. Whenever $\dim(\mathfrak{g}) \in \mathcal{O}(\text{poly}(n))$, 𝔤-sim yields a scalable classical simulation framework for irreps with $\dim(\mathcal{L}_\lambda) \in \mathcal{O}(\text{poly}(n))$. **(c)** Summary of the potential applications of 𝔤-sim. These efficient classical simulations have utility in simulation and optimization of quantum systems, including pre-training of VQA or QML problems, compiling unitary processes to compact quantum circuits, and characterizing QML optimization landscapes.

grows polynomially with the system size n (i.e., $\dim(\mathfrak{g}) \in \mathcal{O}(\text{poly}(n))$). For example, this is known to occur in systems with with permutation symmetries [41, 45], underlying free-fermionic algebras [46, 47] , continuous-variable or free-bosonic systems [48–50] and others [51, 52]. The study and cataloging of the many Lie algebras occurring in quantum systems is an active field of research [52–57].

In this work, we present a framework for the efficient implementation of Lie-algebraic simulations (referred to as 𝔤-sim) and their utilization. As a first contribution, we extend techniques from Refs. [43, 44], significantly widening their scope. We formulate these techniques in a modern language oriented to the quantum computing community, offer new optimizations that improve their efficiency and extend their utility to optimizing circuits (as opposed to merely simulating them). Comparing 𝔤-sim against Wick-based simulation techniques we show that these can be efficient for different types of states and observables, hinting at some deep connections with resource theory [58], and demonstrating that our proposed method enables new classical simulation scenarios beyond those reachable via Wick's theorem. Our second contribution is to showcase the utility of 𝔤-sim in four different tasks pertaining to the development of quantum technologies. These include the study of the optimization landscapes of parametrized quantum circuits, improving on the initializations of such circuits, problems of quantum circuit synthesis, and the training of a binary quantum-phase classifier. A high-level overview of our theoretical framework and applications for this work is provided in Fig. 1.

In the first part of this work (Sec. II) we review the theory grounding Lie-algebraic simulations and provide guidelines for their efficient implementations. The exposition here is aimed to be pedagogical, with an eye towards generic and efficient algorithms. It follows from Refs. [43, 44], and extends them in terms of scope (increased set of initial states and of observables supported together with the ability to deal with certain forms of noise in the circuits) and implementation (including both the ability to simulate and differentiate evolution).

In the second part (Sec. III), we summarize necessary conditions for the scalability of 𝔤-sim and demonstrate its capabilities in non-variational benchmark problems. These include both noiseless and noisy simulations of $n = 200$ qubits systems and will serve us to highlight fundamental differences with established classical simulations techniques.

In the last part (Sec. IV), 𝔤-sim is put into practice in four distinct scenarios. As a warm-up example, we provide a large scale study of the overparametrization phenomenon (Sec. IV A). Then, 𝔤-sim is applied for the purpose of the initialization of (non-classically-simulable) circuits in two different setups (Sec. IV B). Next, tasks of circuit synthesis and dynamical evolution are addressed (Sec. IV C). Finally, as an application in the context of QML, we resort to 𝔤-sim to train a quantum-phase classifier (Sec. IV D). Overall, these examples aim at illustrating the diversity of tasks that can be accomplished via 𝔤-sim, and also showcase some avoidable pitfalls.

II. FRAMEWORK

A. Set-up

Through this article we focus on quantum dynamics realized by means of quantum circuits (i.e., digital quantum computations), but stress that the same principles can equally be applied to continuously driven systems (i.e., analog quantum computations). Also, we note that while exposed in the context of the qubit model, the same principles are readily ported to other models of quantum physics.

With this scope in mind, we will henceforth study a system of n qubits with an associated Hilbert space $\mathcal{H} = (\mathbb{C}^2)^{\otimes n}$ of dimension $d = 2^n$, and we denote as $\mathcal{L}$ the space of operators acting on $\mathcal{H}$. We further consider the case when the dynamics of the quantum system are determined by a unitary operator

$$U(\boldsymbol{\theta}) = \prod_{l=1}^{L} e^{-i\theta_l H_l}\,, \tag{1}$$

specified in terms of angles (or parameters) $\boldsymbol{\theta} = [\theta_1, \dots, \theta_L]$ and Hermitian operators H_l taken from some set of *gate generators* $\mathcal{S}$ of the circuit.

In the following we will be concerned with tasks of simulation and optimization of quantum circuits. For an observable O, an initial state $\rho^{(\mathrm{in})}$, and a circuit $U(\boldsymbol{\theta})$, *simulation* consists of evaluating the expectation value of O for the evolved state $\rho^{(\mathrm{out})}(\boldsymbol{\theta}) \equiv U(\boldsymbol{\theta})\rho^{(\mathrm{in})}U^\dagger(\boldsymbol{\theta})$. That is, by simulation we mean the evaluation of a quantity of the form

$$\langle O(\boldsymbol{\theta})\rangle \equiv \mathrm{Tr}\left[OU(\boldsymbol{\theta})\rho^{(\mathrm{in})}U^\dagger(\boldsymbol{\theta})\right] = \mathrm{Tr}\left[O\rho^{(\mathrm{out})}(\boldsymbol{\theta})\right]. \tag{2}$$

In addition to simulation, one is often interested in actively *optimizing* the system's dynamics to achieve a certain objective. For instance, this is at the core of most problems found in the field of VQAs and QML. These rely on optimizing circuit parameters to decrease the value of a loss function that can be evaluated through expectation values. As an example, in a variational quantum eigensolver (VQE) [59] aiming at preparing the ground state of an Hamiltonian H, this loss would be the expectation value $\langle H(\boldsymbol{\theta})\rangle$. Similarly, in the context of QML, the same circuit unitary may be applied to a dataset of quantum states, and the loss to be minimized could be a function of expectation values of a set of observables $\{O_i\}$ that are obtained for each state. In both cases, optimization of the circuit parameters can be enhanced by the ability to compute gradients of the form $\partial_{\boldsymbol{\theta}}\langle O(\boldsymbol{\theta})\rangle$.

B. Lie theory

Here we recall elementary results of Lie theory that are relevant to this work. For a general treatment of Lie groups and Lie algebras, we recommend standard textbooks [60, 61]. A more specific presentation of Lie theory in the context of quantum control can be found in Refs. [62, 63], or in the context of quantum circuits in Refs. [64, 65]. We start by reviewing the concepts of dynamical Lie algebra (Definition 1) and dynamical Lie group (Definition 2) which characterize the expressiveness of quantum circuits of the form in Eq. (1).

Definition 1 (Dynamical Lie algebra)**.** *Consider an ansatz of the form in Eq. (1). Its dynamical Lie algebra $\mathfrak{g}$ is defined as the vector space spanned by all the possible nested commutators of $i\mathcal{S} = \{iH_1, \dots, iH_L\}$. That is,*

$$\mathfrak{g} = \mathrm{span}_{\mathbb{R}} \langle\{iH_1, \dots, iH_L\}\rangle_{\mathrm{Lie}} \subseteq \mathfrak{su}(2^n)\,, \tag{3}$$

where $\langle\cdot\rangle_{\mathrm{Lie}}$ denotes the Lie closure, i.e., the set of elements obtained by repeatedly taking nested commutators until no new linearly independent elements are obtained. We denote as $\{iG_\gamma\}_{\gamma=1}^{\dim(\mathfrak{g})}$ a set of anti-Hermitian operators which form a Schmidt-orthonormal basis of $\mathfrak{g}$.

The dynamical Lie algebra $\mathfrak{g}$ constitutes a Lie subalgebra of the special unitary Lie algebra $\mathfrak{su}(2^n)$, the space of linear anti-Hermitian operators acting on the n-qubit system. For details on numerical algorithms to compute the Lie closure we refer to Refs. [63, 64, 66].

In correspondence to the dynamical Lie algebra $\mathfrak{g}$, we can also define the dynamical Lie group $\mathcal{G}$ as follows.

Definition 2 (Dynamical Lie group)**.** *The dynamical Lie group $\mathcal{G}$ of a circuit of the form Eq. (1) is determined by its dynamical Lie algebra $\mathfrak{g}$ and defined as*

$$\mathcal{G} = e^{\mathfrak{g}} \equiv \{e^{iA}, \quad iA \in \mathfrak{g}\}\,. \tag{4}$$

Notably, the dynamical Lie group $\mathcal{G}$ corresponds to all possible unitaries that can be implemented by circuits of the form in Eq. (1). That is, for every $V \in \mathcal{G}$ there exists (at least) one choice of parameter values $\boldsymbol{\theta}$ for a sufficiently large, but finite, number of layers L such that $U(\boldsymbol{\theta}) = V$ [62]. Overall, $\mathfrak{g}$ determines the group $\mathcal{G}$ of unitaries that could be realized, and we can take their dimensions in correspondence: $\dim(\mathcal{G})$ (as a smooth differentiable manifold) $= \dim(\mathfrak{g})$.

Having defined the Lie algebra and group associated with the dynamical process in Eq. (1), we can now study how they act on the operator space $\mathcal{L}$. Our strategy to evaluate Eq. (2) will consist in evolving operators – either the state $\rho^{(\mathrm{in})}$ or the observable O – through conjugation by $U(\boldsymbol{\theta})$. Hence we are interested in the action $A \mapsto VAV^\dagger$ of the group unitaries $V \in \mathcal{G}$ onto operators $A \in \mathcal{L}$. For arbitrary V, evaluating such action would scale polynomially with $\dim(\mathcal{L}) = d^2$. However, we can exploit the structure of $\mathcal{G}$ to decompose $\mathcal{L}$ into smaller subspaces in order to simplify the problem. These are the invariant subspaces that are now defined.

Definition 3 (Invariant subspace)**.** *We say that an operator subspace $\mathcal{L}_\lambda \subset \mathcal{L}$ is invariant under the action of $\mathcal{G}$ whenever*

$$VAV^\dagger \in \mathcal{L}_\lambda, \forall A \in \mathcal{L}_\lambda \text{ and } \forall V \in \mathcal{G}. \tag{5}$$

Furthermore we say that $\mathcal{L}_\lambda$ is irreducible (and call it an irrep) when it is invariant and when there exists no non-trivial $\mathcal{L}_\mu \subset \mathcal{L}_\lambda$ that is also invariant.

For compact groups acting on finite dimensional spaces, as considered throughout, one can always decompose $\mathcal{L}$ in terms of invariant subspaces $\mathcal{L}_\lambda$ as

$$\mathcal{L} \cong \bigoplus_\lambda \mathcal{L}_\lambda. \tag{6}$$

The maximal of such decomposition, whereby each $\mathcal{L}_\lambda$ does not further contain invariant subspaces, is called the irreducible (irrep) decomposition. The benefits of such decomposition is that the complexity of evaluating Eq. (2) becomes related to $\dim(\mathcal{L}_\lambda)$ that can be substantially smaller than $\dim(\mathcal{L})$.

In what follows we will generically refer to the $\mathcal{L}_\lambda$ as *the irreps*, but we note that all results hold true provided that $\mathcal{L}_\lambda$ are invariant subspaces, not necessarily irreducible ones. For instance, in the demonstration and application sections of Sec. III and IV, we will focus on the case where $\mathcal{L}_\lambda = i\mathfrak{g}$. Although not irreducible in general, $i\mathfrak{g}$ is an invariant subspace, as shown in Appendix A 1. Furthermore, we will denote as $B_\alpha^{(\lambda)}$ (with $\alpha = 1, \dots, \dim(\mathcal{L}_\lambda)$) the Hermitian operators such that $\{B_\alpha^{(\lambda)}\}_\alpha$ forms a Schmidt-orthonormal basis of $\mathcal{L}_\lambda$. We say that an observable A is *supported* by the irrep $\mathcal{L}_\lambda$ whenever $A \in \mathcal{L}_\lambda$, such that A can be entirely decomposed in the *basis of observables* $\{B_\alpha^{(\lambda)}\}_\alpha$.

The fact that operator space decomposes into irreps allows us to precisely characterize how a given operator transform under the action of a unitary in the dynamical Lie group. For this purpose, we begin by recalling that given an operator $iG_\gamma \in \mathfrak{g}$, its adjoint action on an irrep $\mathcal{L}_\lambda$ is fully characterized by taking a Hermitian basis of $\mathcal{L}_\lambda$ and computing the *representation elements* $f_{\alpha\beta}^{(\lambda)\gamma} = \mathrm{Tr}\Big[iB_\beta^{(\lambda)}\Big[iG_\gamma, iB_\alpha^{(\lambda)},\Big]\Big] \in \mathbb{R}$, through

$$\Big[iG_\gamma, iB_\alpha^{(\lambda)}\Big] = \sum_{\beta=1}^{\dim(\mathcal{L}_\lambda)} f_{\alpha\beta}^{(\lambda)\gamma} iB_\beta^{(\lambda)}\,. \tag{7}$$

By linearity, these constants fully capture the action of any Lie-algebra elements over any operators belonging to $\mathcal{L}_\lambda$, in terms of $\dim(\mathfrak{g})$ matrices of sizes $\dim(\mathcal{L}_\lambda) \times \dim(\mathcal{L}_\lambda)$. We leverage the following definition.

Definition 4 (adjoint representation of $\mathfrak{g}$ in the λ-th irrep)**.** *Given an operator iG_γ in $\mathfrak{g}$, its adjoint representation on the irrep $\mathcal{L}_\lambda$ is obtained via the map $\Phi_\lambda^{\mathrm{ad}} : \mathfrak{g} \mapsto \mathbb{R}^{\dim(\mathcal{L}_\lambda)\times\dim(\mathcal{L}_\lambda)}$ and is defined by*

$$\big(\Phi_\lambda^{\mathrm{ad}}(iG_\gamma)\big)_{\alpha\beta} \equiv f_{\alpha\beta}^{(\lambda)\gamma} = \mathrm{Tr}\Big[iB_\beta^{(\lambda)}\Big[iG_\gamma, iB_\alpha^{(\lambda)}\Big]\Big]\,. \tag{8}$$

We note that if the underlying algebra is compact, and if the basis observables $B_\alpha^{(\lambda)}$ of the irreps are Schmidt-orthonormal, then the adjoint representation is faithful [43] (i.e., the map $\Phi_\lambda^{\mathrm{ad}}$ is injective). It is clear that since the adjoint representation is linear, Definition 4 implies that knowledge of the adjoint representations of the $\dim(\mathfrak{g})$ basis elements is sufficient to obtain the representation of any element of $\mathfrak{g}$. That is, for any $A = \sum_\gamma (\boldsymbol{w})_\gamma G_\gamma$ (with $\boldsymbol{w} \in \mathbb{R}^{\dim(\mathfrak{g})}$) supported by the algebra, one has $\Phi_\lambda^{\mathrm{ad}}(A) = \sum_\gamma (\boldsymbol{w})_\gamma \Phi_\lambda^{\mathrm{ad}}(G_\gamma)$.

Next, by means of the exponential map, we can obtain the adjoint representation of the elements of the Lie group $\mathcal{G}$.

Definition 5 (Adjoint representation of $\mathcal{G}$ in the λ-th irrep)**.** *The adjoint representation of $\mathfrak{g}$ in $\mathcal{L}_\lambda$ induces the adjoint representation $\Phi_\lambda^{\mathrm{Ad}}$ of $\mathcal{G}$. This representation is a linear map $\Phi_\lambda^{\mathrm{Ad}} : \mathcal{G} \mapsto \mathbb{GL}(\mathbb{R}^{\dim(\mathcal{L}_\lambda)}) \subset \mathbb{R}^{\dim(\mathcal{L}_\lambda)\times\dim(\mathcal{L}_\lambda)}$ from the group $\mathcal{G}$ to the group of invertible linear operators $\mathbb{GL}(\mathbb{R}^{\dim(\mathcal{L}_\lambda)})$, defined as*

$$\Phi_\lambda^{\mathrm{Ad}}(U = e^{iA}) = e^{i\bar{\boldsymbol{A}}}\,, \tag{9}$$

for all $iA \in \mathfrak{g}$ (or any $U \in \mathcal{G}$), and with $\bar{\boldsymbol{A}} \equiv \Phi_\lambda^{\mathrm{ad}}(A)$.

The main appeal of such representation is that we can evaluate the action of any $U \in \mathcal{G}$ over any basis element $B_\alpha^{(\lambda)}$ of any $\mathcal{L}_\lambda$ as

$$U^\dagger B_\alpha^{(\lambda)} U = \sum_\beta \big(\Phi_\lambda^{\mathrm{Ad}}(U)\big)_{\alpha\beta} B_\beta^{(\lambda)}\,. \tag{10}$$

We refer the reader to Appendix A 1 for more details on Eq. (10). As further detailed below, these adjoint representations allow us to perform (Heisenberg) evolution of any operator in an irrep $\mathcal{L}_\lambda$ under the adjoint action of any $U \in \mathcal{G}$ with a time complexity scaling with $\mathcal{O}(\mathrm{poly}(\dim(\mathcal{L}_\lambda)))$ as opposed to $\Theta(4^n)$.

Before moving further, we highlight that faithfulness of a Lie algebra representation does not imply faithfulness of its corresponding Lie group representation. To understand this, we need to recall what the center $Z(\mathcal{G})$ of $\mathcal{G}$ is.

Definition 6 (Center of a group)**.** *The center of a group $\mathcal{G}$ is the subset of $\mathcal{G}$ that simultaneously commutes with all elements of $\mathcal{G}$. That is,*

$$Z(\mathcal{G}) \equiv \{W \in \mathcal{G} \mid \forall U \in \mathcal{G}, WU = UW\}\,. \tag{11}$$

Specifically, one can readily verify that for any $U = e^{iA}$ and $W \in Z(\mathcal{G})$, we have that $\forall\lambda$

$$\Phi_\lambda^{\mathrm{Ad}}(WU) = \Phi_\lambda^{\mathrm{Ad}}(UW) = \Phi_\lambda^{\mathrm{Ad}}(U)\,, \tag{12}$$

showing that a non-trivial center $Z(\mathcal{G})$ results in unfaithfulness of the $\Phi_\lambda^{\mathrm{Ad}}$. Even though the Lie algebras $\mathfrak{g}$ considered in this work are centerless, the Lie groups $\mathcal{G} = e^{\mathfrak{g}}$ can still have a nontrivial center, leading to unfaithfulness of $\Phi_\lambda^{\mathrm{Ad}}$. While not an issue for most situations encountered, we will see later in Sec. IV C that this introduces additional considerations for unitary compilation.

C. $\mathfrak{g}$-sim principles

In this section we present the main results which comprise the foundation of $\mathfrak{g}$-sim. These include noiseless simulations with observables supported by one or multiple irreps (Sec. II C 1) or products of observables (Sec. II C 2), noisy simulations (Sec. II C 3), and optimizations (Sec. II C 4). Additional details are provided in App. A, B and C.

1. Simulation with observables supported by a single irrep

Let us first consider the case of simulation problems where the observable of interest is supported by a single irrep. That is, we want to evaluate Eq. (2) when $O \in \mathcal{L}_\lambda$ for some λ.

Let us define $\dim(\mathcal{L}_\lambda)$-dimensional vectors of expectation values that captures the description of the input and output states $\rho^{(\text{in/out})}$ over the basis of observables $B_\alpha^{(\lambda)}$:

$$\left(\boldsymbol{e}_\lambda^{(\text{in/out})}\right)_\alpha \equiv \operatorname{Tr}\left[B_\alpha^{(\lambda)} \rho^{(\text{in/out})}\right]. \tag{13}$$

The following result holds.

Result 1 (Simulation of observables in the λ-th irrep)**.** *Consider a circuit of the form in Eq. (1), and let O be an observable with support in $\mathcal{L}_\lambda$ such that $O = \sum_\alpha (\boldsymbol{w})_\alpha B_\alpha^{(\lambda)}$ for $\boldsymbol{w} \in \mathbb{R}^{\dim(\mathcal{L}_\lambda)}$. Then, given $\boldsymbol{e}_\lambda^{(\text{in})}$, we can compute*

$$\langle O(\boldsymbol{\theta})\rangle = \boldsymbol{w} \cdot \boldsymbol{e}_\lambda^{(\text{out})}, \tag{14}$$

with the vector of output expectation values obtained as

$$\boldsymbol{e}_\lambda^{(\text{out})} = \left(\prod_{l=1}^{L} e^{-i\theta_l \bar{\boldsymbol{H}}_l}\right) \cdot \boldsymbol{e}_\lambda^{(\text{in})}, \tag{15}$$

where $\bar{\boldsymbol{H}}_l \equiv \Phi_\lambda^{\text{ad}}(H_l)$.

Result 1 indicates that, in order to compute $\langle O(\boldsymbol{\theta})\rangle$, it suffices to have a decomposition of O in the Hermitian basis of the irrep, the adjoint representations $\bar{\boldsymbol{H}}_k$ of the gate generators, and the vector of expectation values of the basis observables $\{B_\alpha^{(\lambda)}\}_{\alpha=1}^{\dim(\mathcal{L}_\lambda)}$ for the input state. We stress that although Eq. (15) resembles unitary evolution in the Hilbert space $\mathcal{H}$, it differs subtly. The vectors $\boldsymbol{e}_\lambda^{(\text{in/out})}$ are not state vectors in the usual sense, but vectors of expectation values of observables, and consequently the phases (signs) are indeed physical. Since the $\bar{\boldsymbol{H}}_l$ are purely imaginary Hermitian matrices, the gate representations $e^{-i\theta_l \bar{\boldsymbol{H}}_l}$ are real-valued, and describe linear coupling between observables induced by unitary evolution.

An immediate consequence of Result 1 is that we can compute expectation values of observables supported by the irrep with a time complexity scaling as $\mathcal{O}(L \dim(\mathcal{L}_\lambda)^3)$, with the cubic power arising from matrix exponentiation. However, as detailed in Appendix B 1, we provide a more efficient algorithm based on precomputation of the eigendecomposition of the $\bar{\boldsymbol{H}}_k$ matrices. Further improvements for algebras with a Pauli basis are provided in Appendices B 2 and B 3. This leads us to our first main contribution, which improves on the time complexity of $\mathfrak{g}$-sim.

Theorem 1. *Computing expectation values of observables supported by a given irrep $\mathcal{L}_\lambda$ using $\mathfrak{g}$-sim has a time complexity in $\mathcal{O}(L \dim(\mathcal{L}_\lambda)^2)$ for circuits of the form in Eq. (1).*

Result 1 and Theorem 1 can be naturally extended to expectation values of operators supported in multiple irreps $\mathcal{L}_\lambda$ due to linearity. In particular, we have that the following result holds.

Result 2 (Simulation of operators in multiple irreps.)**.** *Consider a circuit of the form in Eq. (1), and let O be an observables with support in a set of irreps $\{\mathcal{L}_{\lambda_i}\}_i$ such that $O = \sum_i O^{(\lambda_i)}$ where $O^{(\lambda_i)} = \sum_\alpha (\boldsymbol{w}^{(\lambda_i)})_\alpha B_\alpha^{(\lambda_i)}$ for $\boldsymbol{w}^{(\lambda_i)} \in \mathbb{R}^{\dim(\mathcal{L}_\lambda)}$. Then, given the vectors $\boldsymbol{e}_{\lambda_i}^{(\text{in})}$, we can compute*

$$\langle O(\boldsymbol{\theta})\rangle = \sum_i \boldsymbol{w}^{(\lambda_i)} \cdot \boldsymbol{e}_{\lambda_i}^{(\text{out})}, \tag{16}$$

with each vector of output expectation values is obtained as in Eq. (15).

Here, we can see that Theorem 1 directly implies the following corollary.

Corollary 1. *Computing expectation values of an observable with support in a constant set of irreps $\{\mathcal{L}_{\lambda_i}\}_i$ has a time complexity in $\mathcal{O}(L \max_i\{\dim(\mathcal{L}_{\lambda_i})^2\})$.*

2. Simulation with products of observables each supported by a single irrep

Taking a step further, we can simulate expectation values of product of observables each supported by a single irrep. First, let us focus on simulating correlators of the form $\langle O^{(1)} O^{(2)}(\boldsymbol{\theta})\rangle = \operatorname{Tr}\left[O U(\boldsymbol{\theta}) \rho^{(\text{in})} U^\dagger(\boldsymbol{\theta})\right]$ for $O = O^{(1)} O^{(2)}$ with $O^{(1)} \in \mathcal{L}_{\lambda_1}$, and $O^{(2)} \in \mathcal{L}_{\lambda_2}$. For these, we define a $(\dim(\mathcal{L}_{\lambda_1}) \times \dim(\mathcal{L}_{\lambda_2}))$-dimensional matrix of expectation values that captures the description of the states $\rho^{(\text{in/out})}$ over products of the basis observables

$$(\boldsymbol{E}^{(\text{in/out})})_{\alpha\beta} \equiv \operatorname{Tr}\left[B_\alpha^{(\lambda_1)} B_\beta^{(\lambda_2)} \rho^{(\text{in/out})}\right]. \tag{17}$$

As shown in Appendix A 2, we have the following result.

Result 3 (Simulation of product of two operators)**.** *Consider a circuit of the form in Eq. (1), and let $O^{(1)}$ and $O^{(2)}$ be observables with support in $\mathcal{L}_{\lambda_1}$ and*

$\mathcal{L}_{\lambda_2}$ respectively such that $O^{(1)} = \sum_\alpha (\boldsymbol{w}^{(1)})_\alpha B_\alpha^{(\lambda_1)}$ and $O^{(2)} = \sum_\beta (\boldsymbol{w}^{(2)})_\beta B_\beta^{(\lambda_2)}$ for $\boldsymbol{w}^{(1)} \in \mathbb{R}^{\dim(\mathcal{L}_{\lambda_1})}$ and $\boldsymbol{w}^{(2)} \in \mathbb{R}^{\dim(\mathcal{L}_{\lambda_2})}$. Then, given $\boldsymbol{E}^{(\text{in})}$, we can compute

$$\langle O^{(1)} O^{(2)}(\boldsymbol{\theta})\rangle = (\boldsymbol{w}^{(1)})^T \cdot \boldsymbol{E}^{(\text{out})} \cdot \boldsymbol{w}^{(2)}\,, \tag{18}$$

with the superscript T denoting the matrix transpose, and with the matrix of output expectation values obtained as

$$\boldsymbol{E}^{(\text{out})} = \left(\prod_{l=1}^{L} e^{-i\theta_l \bar{\boldsymbol{H}}_l}\right) \boldsymbol{E}^{(\text{in})} \left(\prod_{l=1}^{L} e^{-i\theta_l \bar{\boldsymbol{H}}_l}\right)^T. \tag{19}$$

The complexity of such simulations is provided in the following Corollary, and we note that it is the same as Corollary. 1.

Corollary 2. *Computing expectation values of products of observables $O = O^{(1)}O^{(2)}$ with $O^{(1)} \in \mathcal{L}_{\lambda_1}, O^{(2)} \in \mathcal{L}_{\lambda_2}$ has a time complexity in $\mathcal{O}(L \max_i\{\dim(\mathcal{L}_{\lambda_i})^2\})$.*

Going further, $\mathfrak{g}$-sim can be extended to compute expectation values of correlators of any orders, such as a M-th order product $O^{(1)} \dots O^{(M)}$ with each $O^{(m)}$ supported by potentially distinct $\mathcal{L}_\lambda$. However, as detailed in Appendix A 2, simulating an M-th order correlator for arbitrary initial state has complexity $\mathcal{O}(\max_i\{\dim(\mathcal{L}_{\lambda_i})^M\})$ that becomes impractical for large-order M. In this work, however we require only $M \leqslant 2$.

Finally, we note that, in general, a given observable supported by an irrep $\mathcal{L}_\lambda$ can also be decomposed as a product of two or more observables supported in different irreps. Hence, depending on the decomposition adopted, its simulation could be performed through Result 1 or through the results of this section. Different approaches entail different computational scalings such that, in implementations, one may be favored to the other.

3. Noisy simulations

From the previous discussion, one can already see how some restricted forms of noise can be introduced in the simulations. Over– and under– rotations in the circuits gates of Eq. (1) are readily captured through updates of the angles $\theta_l \to \theta_l + \varepsilon_l$ by some perturbations ε_l. Fluctuations or uncertainty in such angles are modeled by repeatedly sampling perturbations ε_l and averaging over the different realizations. Going further, one can also incorporate noise channels consisting of probabilistic occurrences of elements of $\mathcal{G}$. All these, however, may be rather limited as we are forced to consider noise that is generated by $\mathfrak{g}$, and introduce some overhead due to repeated sampling of the noise realizations. However, as we now discuss (and detail further in App. A 3), often we can do much more and do it more efficiently.

To capture the noise that can be simulated in $\mathfrak{g}$-sim, we introduce the notion of the normalizer of an irrep.

Definition 7 (Normalizer of $\mathcal{L}_\lambda$)**.** *Given an irrep $\mathcal{L}_\lambda$, we define its normalizer as all the operators that leaves the subspace invariant under conjugation:*

$$\mathrm{N}(\mathcal{L}_\lambda) := \{A \in \mathcal{L} \,|\, A^\dagger X A \in \mathcal{L}_\lambda,\, \forall X \in \mathcal{L}_\lambda\}. \tag{20}$$

We stress that by definition of the irreps, as per Definition. 3, we have $\mathcal{G} \subset \mathrm{N}(\mathcal{L}_\lambda)$ for any $\mathcal{L}_\lambda$. However, the normalizer may contain many more operators $A \notin \mathcal{G}$. Notably, whenever the Lie algebra $\mathfrak{g}$ has a basis of Pauli operators, any of the exponentially many Pauli operators will belong to the normalizer, and that independently of $\dim(\mathfrak{g})$. As we will soon see, this means that action of Pauli noise channels can be dealt with in $\mathfrak{g}$-sim. A demonstration of such possibilities is provided in Sec. III.

Let us consider a noise channel Λ with decomposition

$$\Lambda[\cdot] = \sum_k E_k \cdot E_k^\dagger, \tag{21}$$

in terms of Kraus operators E_k satisfying $\sum_k E_k^\dagger E_k = I$. Whenever $E_k \in \mathrm{N}(\mathcal{L}_\lambda)$ *for all* k, and if $O \in \mathcal{L}_\lambda$, we have the guarantee that $\Lambda[O] \in \mathcal{L}_\lambda$. Akin to Definition 5, we can define the adjoint representation of any $A \in \mathrm{N}(\mathcal{L}_\lambda)$, and by linearity of Λ, as $\dim(\mathcal{L}_\lambda) \times \dim(\mathcal{L}_\lambda)$ matrices. The latter fully captures the action of Λ on $\mathcal{L}_\lambda$: given the adjoint representation $\Phi_\lambda^{\text{Ad}}(\Lambda)$ of a channel Λ and an observable $O = \sum_\alpha (\boldsymbol{w})_\alpha B_\alpha^{(\lambda)} \in \mathcal{L}_\lambda$, we get that $\Lambda(O) = \sum_\alpha (\boldsymbol{w}')_\alpha B_\alpha^{(\lambda)}$ with updated weights $\boldsymbol{w}' = \Phi_\lambda^{\text{Ad}}(\Lambda) \cdot \boldsymbol{w}$.

In noisy simulations with $\mathfrak{g}$-sim adjoint representations of the noise channels are simply interleaved in between adjoint representations of the unitary gates and does only incur constant overhead in the simulations. This is formalized in Result 4 of Appendix A 3 a and captured by the following corollary:

Corollary 3. *Provided that the noise channels have a Kraus decomposition as per Eq. (21) such that all $E_k \in \mathrm{N}(\mathcal{L}_\lambda)$, then Corrolary 1 extends to exact noisy simulations with the same complexities.*

We see that whenever the noise channels admit a Kraus decompositions with operators supported by $\mathrm{N}(\mathcal{L}_\lambda)$, we can perform *exact* noisy simulations for observables with support in a constant set of irreps, with the same time and memory complexity as the noiseless simulations. This is in stark contrast with typical noisy simulations of quantum systems that are either approximate (based on sampling many random realizations of the noise), or exact but incurring a quadratic increase in memory and computing requirements. However, as detailed In Appendix. A 3 b, the case of noisy simulations for product of observables needs to be performed though sampling of noise trajectories. Given S of such trajectories, this incurs a complexity $\mathcal{O}(S \max_i\{\dim(\mathcal{L}_{\lambda_i})^2\})$ for an error scaling as $\mathcal{O}(\sqrt{S}^{-1})$.

4. Simulation of gradients

In addition to simulating parametrized quantum circuits, in a variational quantum computing setting one aims to *optimize* the parameters to minimize a loss function. In order to benefit from gradient-based training schemes one needs to compute derivatives of expectation values with respect to the circuit parameters.

Given that Lie-algebraic techniques were originally envisioned for simulating fixed unitary dynamics and not for their optimization, there are no existing methods that use $\mathfrak{g}$-sim to compute partial derivatives. However, due to the form of the evolution in the adjoint representation of Eq. (15), we can port reverse-mode differentiation methods [67] to $\mathfrak{g}$-sim yielding an efficient algorithm for gradient computation. Such an algorithm is presented in detail in Appendix C 1 and, here, we only remark on the complexity entailed.

Theorem 2 (Gradient calculations in $\mathfrak{g}$-sim)**.** *Computing the gradient of the expectation value of an observable supported by an irrep $\mathcal{L}_\lambda$ using* $\mathfrak{g}$-sim *has a time complexity in $\mathcal{O}(L\dim(\mathcal{L}_\lambda)^2)$ for circuits of the form in Eq.* (1).

Overall, the previous results form the basis of $\mathfrak{g}$-sim comprising noisy and noiseless simulations and optimizations of quantum ciruits. In Appendices A 4 and C 3 we detail how this is extended to analog quantum computing. In the following we demonstrate situations where $\mathfrak{g}$-sim leads to scalable computations when we specialize to the case where $\mathcal{L}_\lambda = i\mathfrak{g}$.

III. SCALABILITY AND COMPARISON

In this section, we aim to demonstrate the capabilities of $\mathfrak{g}$-sim and at contrasting it to other established simulation techniques. In Sec. III A, we collect and summarize conditions for scalability. In Sec. III B, we contrast $\mathfrak{g}$-sim to simulation techniques based on Wick's theorem, including the standard treatment of free-fermion systems. In Sec. III C, we introduce a Lie algebra of interest $\mathfrak{g}_0$. This Lie algebra will form the basis of both the numerical examples of this section and of many of the tasks tackled later on in the application section (Sec. IV). In Sec. III D we report results for noiseless and noisy simulations for system sizes of $n = 200$ qubits and discuss distinction of $\mathfrak{g}$-sim compared to related free-fermion methods. Additionally, we present details of a benchmark test to numerically verify polynomial resource scalings.

A. Scalability of $\mathfrak{g}$-sim

As discussed in the previous section, $\mathfrak{g}$-sim recasts quantum evolution as linear algebra problems on vectors in $\mathbb{R}^{\dim(\mathcal{L}_\lambda)}$ and matrices in $\mathbb{R}^{\dim(\mathcal{L}_\lambda)\times\dim(\mathcal{L}_\lambda)}$, as well as linear combinations thereof. Although such techniques can be applied to *any* system, most dynamical Lie algebras, and their associated irreps, have dimension scaling as $\Theta(4^n)$ (e.g., randomly sampled anti-Hermitian operators generate the whole special unitary algebra $\mathfrak{su}(d)$ [68]), and thus this framework does not in general yield an asymptotic advantage. Yet, as pointed out earlier, special cases exist where such dimension scales as $\mathcal{O}(\mathrm{poly}(n))$, such that $\mathfrak{g}$-sim can be used to perform classically efficient simulations despite the exponential dimension of $\mathcal{H}$.

For convenience, we explicitly reiterate the conditions required for classically efficient simulations via $\mathfrak{g}$-sim.

1. The Lie closure of the gate generators must lead to an algebra $\mathfrak{g}$ such that there exists irreps $\mathcal{L}_\lambda$ with $\dim(\mathcal{L}_\lambda) \in \mathcal{O}(\mathrm{poly}(n))$.

2. We must know a Schmidt-orthonormal basis of the polynomially-large $\mathcal{L}_\lambda$, as well as the non-zero representation elements.

3. Observables of interest must be supported by polynomially-large irreps, or products of terms, each supported by polynomially-large irreps, up to some constant order M.

4. The expectation values of the irreps basis elements (or their products up to some fixed order M), over the initial state must be known.

Let us make some brief comments regarding these requirements. First, we recall that while most evolutions lead to exponentially sized algebras, there exists examples where the algebra grows polynomially with the number of qubits [41, 45–51], including free-fermions, free-bosons systems with permutation symmetries and more. We further note that studies of Lie-algebra, originally motivated by problems of controllability, have received renewed attention lately. While recent works [52, 55–57], have been mostly limited to qubit systems and to Lie algebras generated by Pauli operators, we expect that it would extend to more families of Lie-algebras. In turn, these could offer many more applications for the techniques presented.

B. Comparison of $\mathfrak{g}$-sim to other Wick-based simulation methods; connections to resource theory

At this point, we find it convenient to briefly compare $\mathfrak{g}$-sim to a different Lie-algebraic simulation method: Wick's theorem. Commonly, the latter is used to simulate non-interacting free-fermionic evolutions [69], i.e., unitaries generated by Hamiltonians expressed as a combination of quadratic products of fermion creation and annihilation operators. We refer the reader to Appendix D, but also to Refs. [43] for additional details on this method. In particular, one can show that there exist polynomially-sized Lie algebras $\mathfrak{g}$ for which Wick's theorem enables the efficient classical simulation of expectation values for a O in any $\mathcal{L}_\lambda$ that is expressed as a product of polynomially-many products of elements in $i\mathfrak{g}$ [3], so long as the initial state is either a generalized coherent state [70–73] (i.e., $\rho^{(\mathrm{in})} = V\,|\mathrm{hw}\rangle\langle\mathrm{hw}|\,V^\dagger$ for $V \in e^{\mathfrak{g}}$ and $|\mathrm{hw}\rangle$ the *highest weight state* of $\mathfrak{g}$), or a linear combination of polynomially-many generalized coherent state. As such, simulation techniques based on Wick's theorem will become exponentially expensive when the initial state has to be decomposed into an exponential number of generalized coherent states. On the other hand, when using $\mathfrak{g}$-sim, the simulation is efficient for general states (provided we can get the associated expectation values in the basis of $\mathcal{L}_\lambda$) but for observables in irreps whose dimension scales only polynomially with the system size, as per Theorem 1. Broadly speaking, we can summarize these results as the following constraints:

1. Wick-based simulation: General observable, special state.

2. $\mathfrak{g}$-sim-based simulation: General state, special observable.

The previous distinction between Wick-based techniques and $\mathfrak{g}$-sim shows that the two methods have merit for different types of states. On the one hand, Wick-based simulations fail for states with exponential "extent" [74–76], or, in the language of resource theory [58], for states that are expressed as an exponential combination of free states. On the other hand, if the input state has zero component on the poly-size dimensional irreps, then its dynamics therein are trivial (i.e., $\mathfrak{g}$-sim would predict zero for all observables taken from such irreps). Hence, to perform non-trivial simulations, one would have to implement $\mathfrak{g}$-sim on the exponentially large irreps, in which case it becomes inefficient. In fact, it has been recently proposed that the irrep decomposition of a quantum state is a measure of its resourcefulness [77], and that highly-resourceful states have little-to-no component in the smaller irreps. Crucially, since this notion of resourcefulness is non-equivalent to the extent [77] (i.e., states with both order-one and exponentially large extent can fail to have component in the smaller dimensional irreps), we can see that the distinction between Wick-based techniques and $\mathfrak{g}$-sim appears to arise on a more fundamental level — both methods allow for efficient simulations for low-resource states, with the nature of the resource quantified differently for each method. Hence, $\mathfrak{g}$-sim can enable new scenarios for simulation beyond those achievable with Wick-based techniques.

C. A polynomially-large Lie-algebra

As a specific example of an algebra with $\dim(\mathfrak{g}) \in \mathrm{poly}(n)$, we consider a special model with free-fermion mappings where $\dim(\mathcal{L}_{\mathfrak{g}}) \in \mathcal{O}(n^2)$. This algebra, or its subalgebras, will be used in the numerical simulations performed. The algebra under consideration is given by

$$\mathfrak{g}_0 = \mathrm{span}_{\mathbb{R}}\left\langle \left(\bigcup_{\mu,\nu=x,y} \{i\sigma_j^\mu \sigma_{j+1}^\nu\}_{j=1}^{n-1} \right) \cup \{i\sigma_j^z\}_{j=1}^{n} \right\rangle_{\mathrm{Lie}}, \quad (22)$$

with σ_j^α denoting a single-qubit Pauli operator labelled by $\alpha \in \{x, y, z\}$ acting on the qubit i. A basis for $\mathfrak{g}_0$ is given by the set of Paulis [47]

$$i\{\sigma_j^z, \widehat{\sigma_i^x\sigma_j^x}, \widehat{\sigma_i^y\sigma_j^y}, \widehat{\sigma_i^x\sigma_j^y}, \widehat{\sigma_i^y\sigma_j^x} \,|\, 1 \leqslant i < j \leqslant n\}, \quad (23)$$

where we used the notation $\widehat{A_iB_j} = A_i\sigma_{i+1}^z \cdots \sigma_{j-1}^z B_j$. This algebra is a representation of $\mathfrak{so}(2n)$, and hence its dimension is $\dim(\mathfrak{g}_0) = n(2n-1)$ [47, 78]. All the representation elements can be obtained with a time complexity of $\mathcal{O}(n^5)$; in the GitHub repository of Ref. [79] we have reported these representation elements together with the basis.

Expectation values of the basis elements in Eq. (23) (or their products) can be efficiently computed on a classical computer for many families of initial states. These include the highest weight states [43], any product states or stabilizer states, and even superpositions of polynomially many of these. Interestingly, as we explore in the following section, these also include initial states that are magic states for the class of circuits under consideration, allowing $\mathfrak{g}$-sim to efficiently evolve initial states that typical free-fermionic methods could not support — even some with exponentially large fermionic extent. One could even envision the situation where a generic input state is provided as a physical quantum state or as a description of its preparation. One would then estimate these expectation values by using a quantum computer, preparing or accessing $\rho^{(\mathrm{in})}$, and making measurements of all the irrep basis elements. This procedure yields a form of "classical description" [41] of the input state that $\mathfrak{g}$-sim can use.

D. Demonstration and resource benchmark of $\mathfrak{g}$-sim

To demonstrate the power of $\mathfrak{g}$-sim, we first showcase its application to a quantum simulation task which, to the best of our knowledge, *would not be possible with any other known classical simulation method.* We seek to simulate the dynamics of the transverse field XY (TFXY) model with random magnetic fields, whose Hamiltonian reads

$$H_{\mathrm{TFXY}} = \sum_{j=1}^{n-1}(\sigma_j^x\sigma_{j+1}^x + \sigma_j^y\sigma_{j+1}^y) + \sum_{j=1}^{n} b_j\sigma_j^z, \quad (24)$$

where the coefficients b_j are randomly drawn from a Gaussian distribution $N(0, \xi^2)$. We note that Hamiltonian dynamics in $\mathfrak{g}$-sim can be implemented either by directly exponentiating the adjoint representation of the Hamiltonian (9) or by simulating an appropriate quantum circuit (15). Here, we apply the latter as it will allow us to insert noise channels in between gates for subsequent noisy demonstration. The circuit is obtained resorting to a first-order Trotterization of the dynamics with a total of 300 Trotter steps with stepsize $\Delta t = 2$.

For our simulations, we initialize the system in the magic state

$$|\psi\rangle = \left[\frac{|0000\rangle + |0011\rangle + |1100\rangle + e^{i\tau}\,|1111\rangle}{2}\right]^{\otimes \frac{n}{4}}, \quad (25)$$

for $n = 200$ qubits, and aim at computing the dynamics of observables $O \in \mathcal{L}_\lambda = i\mathfrak{g}_0$. Although this state has exponential fermionic extent (i.e., it can only be decomposed into an exponential number of Gaussian states) [80, 81], the vector of expectation values needed for $\mathfrak{g}$-sim can be trivially classically computed, such that simulations are scalable. This highlights the differences between $\mathfrak{g}$-sim and Wick-based methods, as using Wick's theorem along with the state in Eq. (25) would lead to an exponential computational complexity, whereas the cost of $\mathfrak{g}$-sim remains polynomial. Hence, this is an example of a simulation that can be performed efficiently with $\mathfrak{g}$-sim, but not with closely related methods. However, we note that such simulations with $\mathfrak{g}$-sim were possible as we limited ourselves to observables supported by a polynomially-large irrep.

In Figure 2(a) we explore the dynamics of the system in the absence of noise. In particular we explore the propagation of correlations across the system by evaluating the $\langle\sigma_j^y\sigma_{j+1}^x\rangle$ correlators at different evolution times $t = 0, \ldots, 600$. As can be seen, over the time scales probed the correlations propagate to the end of the chain, albeit at small values. Resolving these values is possible due to the exact nature of the simulations performed. The entire simulation was completed in 16 minutes on a single CPU core, demonstrating scalability of $\mathfrak{g}$-sim.

For the Figure 2(b), we repeat this simulation under the presence of noise using the methods outlined in Section II C 3. Given the irrep of interest, we readily see that any Pauli operator belongs to the normalizer $\mathrm{N}(\mathcal{L}_\lambda = i\mathfrak{g}_0)$ as per Definition. 7. Notably, albeit of polynomial dimension, $\mathcal{L}_\lambda$ admits a normalizer containing an exponential number of linearly independent elements. In turn, this allows us to simulate arbitrary Pauli noise channels. In our numerical explorations, following every entangling gate, we apply a random 2-qubit Pauli noise channel

$$\rho \to (1-p)\rho + \sum_{\substack{\nu_1,\nu_2 \in \{1,x,y,z\} \\ (\nu_1,\nu_2)\neq(1,1)}} w_{\nu_1,\nu_2}\sigma_j^{\nu_1}\sigma_k^{\nu_2}\rho\;\sigma_k^{\nu_2}\sigma_j^{\nu_1} \quad (26)$$

where the w_{ν_1,ν_2} are drawn from a uniform distribution $\mathcal{U}(0,1)$ and then normalized such that they sum to p. For the simulation depicted here, we choose an overall per-gate fault probability of $p = 3\times10^{-4}$. Under the presence of noise, the correlations dissipate, although are not entirely eliminated: oscillations can still be seen at the far end of the chain, albeit 6 orders of magnitude smaller than for the noiseless case.

These results highlight the exact nature of noisy simulations with $\mathfrak{g}$-sim: Any method relying on sampling noise realizations would have required of the order of 10^{11} many repetitions to resolve such oscillations. Similarly, classical simulations of noisy dynamics involving the truncation of low-weight contributions in the Heisenberg picture [82, 83] also requires simulating many trajectories.

With a non-trivial application of $\mathfrak{g}$-sim demonstrated, we now consider its performance and scalability more systematically with a benchmarking task, aiming to verify its polynomial resource scaling numerically. We take the circuits $U(\boldsymbol{\theta})$ to be composed of a single layer of gates generated by the set of operators $\left(\bigcup_{\mu,\nu=x,y}\{\sigma_j^\mu\sigma_{j+1}^\nu\}_{j=1}^{n-1}\right) \cup \{\sigma_j^z\}_{j=1}^{n}$. In Fig. 3(a) we schematically show the circuit for $n = 3$.

For the benchmarking task, we evaluate expectation values of random observables in the algebra $\mathfrak{g}_0$ for the state obtained by applying $U(\boldsymbol{\theta})$, with parameters uniformly sampled in the interval $[0, 2\pi]$, to an input state $\rho^{(\mathrm{in})} = |0\rangle\langle 0|^{\otimes n}$. As was the case for the state in Eq. (25), we can easily compute expectation values of $\rho^{(\mathrm{in})}$ for any of the Pauli operators forming the basis of $\mathfrak{g}_0$.

In Fig. 3(b) we show benchmark results in terms of memory and compute time for $\mathfrak{g}$-sim and for full state-vector simulations methods. The benchmarks are computed on a single CPU core - using TensorFlow Quantum for state vector methods, and our Python implementation for $\mathfrak{g}$-sim. Therein we can see that both memory and compute time scale as $\Omega(2^n)$ for state vector methods, and as $\mathcal{O}(\mathrm{poly}(n))$ for $\mathfrak{g}$-sim. The exponential scaling of state-vector simulations makes simulations with $n \geqslant 30$ qubits intractable on the device used. However, using $\mathfrak{g}$-sim we are able to simulate systems of up to 200 qubits on a single core of a modern CPU. Further comments on the benchmarking procedure are given in Appendix E 1.

These demonstrations highlight the capabilities and scalability of our framework for simulations, and in particular its differences and advantages relative to similar methods. These did not however require any optimization; in the next section, we make use of optimization and explore applications of $\mathfrak{g}$-sim to more sophisticated key problems in variational quantum computing.

IV. APPLICATIONS

Having presented the framework for $\mathfrak{g}$-sim, we now explore its benefits in concrete problems. These include a study of the landscape of VQAs highlighting the overparametrization phenomenon (Sec. IV A), designing improved initialization of quantum circuits parameters (Sec. IV B), problems of circuit synthesis (Sec. IV C), and demonstration of the training of a quantum-phase classifier (Sec. IV D). These illustrate the broad range of applications that can be addressed with $\mathfrak{g}$-sim.

We note that in all cases, we will implement $\mathfrak{g}$-sim based on either $\mathfrak{g}_0$ or subalgebras $\mathfrak{g} \subseteq \mathfrak{g}_0$ obtained by removing some of the generators from Eq. (22). In each section, we will specify which subalgebra of $\mathfrak{g}_0$ we will be working with, and a more detailed outline of the relevant subalgebras is given in Appendix F. We further remark

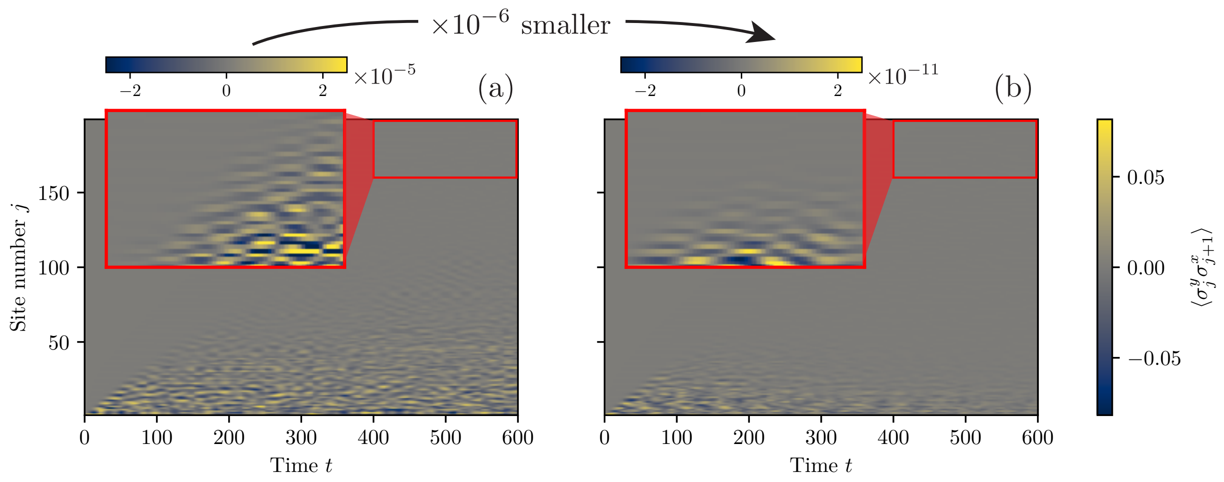


FIG. 2. **Noiseless and noisy dynamics of a 200-qubit magic state using $\mathfrak{g}$-sim.** We simulate the Trotterized dynamics of an initial magic state (25) under a TFXY spin-chain Hamiltonian (24), for $n = 200$ qubits and $\tau = 2.81$ in $\mathfrak{g}$-sim. We report values for correlators of the form $\langle \sigma_j^y \sigma_{j+1}^x \rangle$ along the dynamics. **(a)** In the absence of noise, we see that the correlations propagate across the whole chain. This simulation ran in 16 minutes on a single CPU core. **Inset:** Small oscillations in the correlators reach the far end of the chain, confirming that the light cone of the dynamics reaches from the first to last qubit. **(b)** Including noise in the dynamics, in the form of random 2-qubit Pauli channels (26) acting after each entangling gate with a fault probability of $p = 3 \times 10^{-4}$, we see that the dissipation weakens the correlations by 6 orders of magnitude.

that by definition, we will always be simulating circuits composed of single- and two-qubit gates with local connectivity - that is, circuits that could be implemented on most quantum hardware without incurring large compilation overhead. Details of these circuits are summarized in Table I of Appendix E.

A. Characterizing VQAs

VQAs aim to enable near-term quantum advantage by means of a hybrid quantum-classical training loop, where some of the problem difficulty is offloaded to an optimization problem on a classical co-processor. However, this optimization problem is difficult in general [84], and much remains to be learned about its scalability. Analytic study of optimization landscapes is difficult and limited in scope, while computational study is hindered by the exponential resource costs of state vector simulation. In this context, scalable classical simulations allow one to dequantize certain architectures [32], but also to characterize trainability of VQAs at system sizes that would otherwise be intractable. A previous study has used free-fermion simulations for this purpose [20], but using $\mathfrak{g}$-sim expands the set of systems that can be studied. As a warm-up problem, we demonstrate the onset of overparametrization in VQE problems for the TFXY model at up to $n = 50$ qubits.

1. *Overparametrization in VQE*

Overparametrization [85] is a surprising phenomenon in classical neural networks, where training a neural network with a capacity larger than that which is necessary to represent the training data distribution may lead to improved performance [86–89] and even provable convergence results [90, 91], rather than to the overfitting and training difficulties one may naïvely expect. This phenomenon was generalized to quantum circuits in Ref. [65], where it was shown that the model capacity of circuits of the form Eq. (1) can be quantified by $\dim(\mathfrak{g})$. In turn, overparametrization is characterized by a 'computational phase transition' happening at a critical number of parameters $N_p^{(\mathrm{crit})} \leqslant \dim(\mathfrak{g})$, below which the circuit is hard to train and above which it becomes easy to train. Exact values of this critical threshold $N_p^{(\mathrm{crit})}$ are state-dependent, and often hard to assess analytically as they rely on the conjugation relationship between the initial state and the Lie algebra $\mathfrak{g}$ of the circuit. Thus, exact details of this phenomenon are best probed numerically. However, initial numerical demonstrations of the phenomenon were only provided for systems of 2-10 qubits [65]. Here we demonstrate the phenomenon in problems of up to 50 qubits.

2. *Simulation results*

To probe overparametrization, we consider a VQE task where the goal is to prepare the ground state of the TFXY model of Eq.(24). As an ansatz for $U(\boldsymbol{\theta})$ we

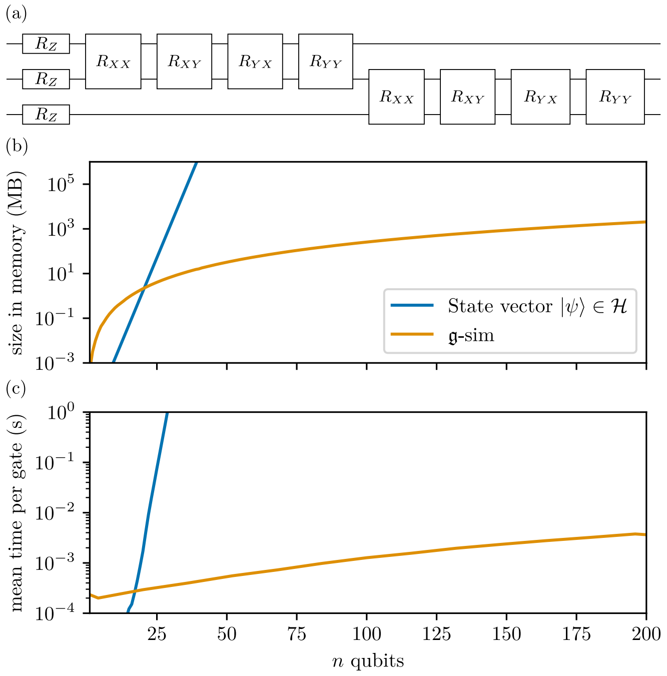


FIG. 3. **Performance benchmarks of $\mathfrak{g}$-sim.** (a) Circuit used for the benchmark at $n = 3$ qubits. The set of generators are presented in Eq. (22). Here, R_Z denotes a rotation about the Z axis, and $R_{\mu\nu}$ a rotation generated by the Pauli operator $\sigma^\mu\sigma^\nu$. We compare the memory (b) and compute time (c) requirements for state vector simulations (blue) and $\mathfrak{g}$-sim (orange) for the unitary shown in panel (a).

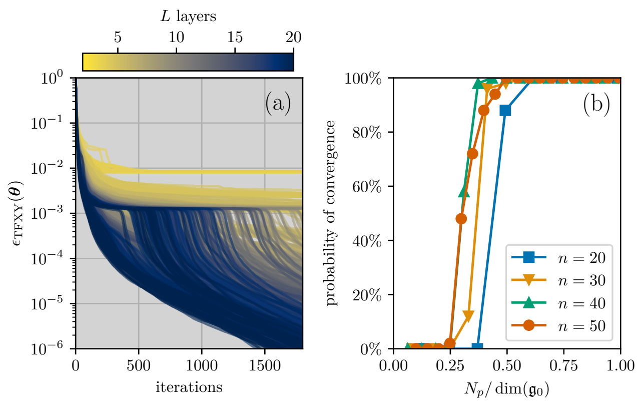


FIG. 4. **Overparametrization in large system sizes.** **(a)** Convergence traces at $n = 50$ qubits of the approximate training error $\epsilon_{\text{TFXY}}(\boldsymbol{\theta})$. **(b)** The probability of converging to $\epsilon_{\text{TFXY}} < 10^{-4}$ for uniform random initialization of $\boldsymbol{\theta}$ as measured by 50 samples, at varied circuit depths, with corresponding number of parameters N_p reported as a fraction of $\dim(\mathfrak{g}_0)$, and for n ranging from 20 to 50 qubits.

consider a Hamiltonian variational ansatz [66, 92] of the form in Eq. (1) with gate generators taken from the set $\{\sigma_j^x\sigma_{j+1}^x, \sigma_j^y\sigma_{j+1}^y\}_{j=1}^{n-1} \cup \{\sigma_j^z\}_{j=1}^{n}$. We refer the reader to Appendix E for additional details on this circuit. One can verify that the dynamical Lie algebra associated with this set of generators is precisely $\mathfrak{g}_0$ in Eq. (22), such that we can use the representation elements reported in Ref. [79]. Given that $\dim(\mathfrak{g}_0) = n(2n-1)$ and that the gates in $U(\boldsymbol{\theta})$ can be parallelized, then overparametrization is achievable with linear circuit depth. To solve the VQE problem, the parameters are optimized using the L-BFGS algorithm with the gradient evaluated according to Appendix C 1 to minimize the energy

$$E_{\text{TFXY}}(\boldsymbol{\theta}) = \langle \mathbf{0}| U^\dagger(\boldsymbol{\theta}) H_{\text{TFXY}} U(\boldsymbol{\theta}) |\mathbf{0}\rangle . \qquad (27)$$

Note that the measurement operator H_{TFXY} is, by definition, fully supported within $\mathfrak{g}_0$. Moreover, since the initial state is the all zero state we can readily construct the vector $\boldsymbol{e}^{(\text{in})}$. That is, we have all the ingredients for $\mathfrak{g}$-sim.

For each of the system sizes probed, from $n = 20$ to 50 qubits, we perform optimization over circuits with a varied number of layers L chosen to result in a number N_p of circuit parameters spanning the range $[1, \dim(\mathfrak{g}_0)]$, and random magnetic field amplitude $\xi = 0.1$. For any of the circuits studied, optimizations are repeated over 50 randomized initial parameter values and fields. Results of this study are reported in Fig. 4.

For a fixed system size of $n = 50$ qubits, we display in Fig. 4(a) all the optimization traces. We observe the convergence behavior of the approximate training error $\epsilon_{\text{TFXY}}(\boldsymbol{\theta}) = (E_{\text{TFXY}}(\boldsymbol{\theta}) - E_{\min})/\|H_{\text{TFXY}}\|_{\text{HS}_1}$, where $E_{\min}$ is the lowest energy discovered by VQE across all depths and $\|\cdot\|_{\text{HS}_1} = \|\cdot\|_{\text{HS}}/2^n$ for $\|\cdot\|_{\text{HS}}$ the Hilbert-Schmidt norm (i.e., we normalize the error relative to the energy scale of the Hamiltonian). Optimization traces exhibit a clear change in the hardness of the optimization problem as the depth of the circuits is varied. Below a critical threshold of $L \approx 6$ layers, we can see severe trainability issues where none of the optimization manages to converge to the ground state energy. However, for circuits with $L > 6$ layers a sudden transition to trainability is observed and solutions converge towards the minimum energy. This phase transition in computational complexity indicates the onset of overparametrization.

More systematically, in Fig. 4(b), we study this phenomenon across varying system sizes n, from 20 to 50 qubits, and report the probability of convergence to $\epsilon_{\text{TFXY}} < 10^{-4}$ as a function of the number of circuit parameters in units of $\dim(\mathfrak{g}_0)$. We observe that the transition to the trainable region (large convergence probability) occurs consistently across all system sizes at $N_p^{(\text{crit})} \approx 0.3 \dim(\mathfrak{g})$ irrespective of n. This supports the analytic results of Ref. [65], and demonstrates the phenomenon of overparametrization at system sizes beyond what could be simulated with state vector simulations. Overall, such detailed analysis of VQA trainability at scale, that requires repetitions over many optimizations and many circuit sizes, is made possible by $\mathfrak{g}$-sim.

Overparametrization is crucial throughout the remainder of this work. It is provably achieved for circuits with $N_p \geqslant \dim(\mathfrak{g})$ parameters, and thus the tractably overparmeterizable models are those with $\dim(\mathfrak{g}) \in \mathcal{O}(\text{poly}(n))$. In cases where the relevant loss function depends entirely on observables supported by the algebra of the circuit (e.g., VQE with Hamiltonian variational ansatz [66, 92–94] or QAOA), circuits overparametrizable in polynomial depth *are exactly those that can be efficiently simulated with* $\mathfrak{g}$*-sim*. Hence, this phenomenon ensures that for any problem that is efficiently simulable in $\mathfrak{g}$-sim, we can guarantee good trainability using a circuit of only polynomial gate-count. Crucially, this ensures a correctly trained prior model for our pre-training scheme (Sec. IV B), the ability to efficiently approximately compile unitaries in $\mathcal{G}$ to linear-depth circuits (Sec. IV C), and good trainability of QML models (Sec. IV D).

B. Pre-training quantum circuits

While the overparametrization regime guarantees trainability from *arbitrary* initial parameter values, in general cases where $\dim(\mathfrak{g}) \in \Theta(4^n)$ (i.e., cases where classical simulations are not possible anymore) it does not result in a scalable strategy. Indeed, trying to access the overparametrized regime would require constructing and training exponentially deep circuits, which is intractable for large problem sizes. Still, it is hoped that with an *adequate* choice of initial parameters one could train the circuits before the onset of overparametrization. Moreover, issues of barren plateaus [13, 64, 95, 96] (another crucial aspect of trainability) indicate exponentially vanishing gradients *on average*, but do not imply that the entire optimization landscape is flat. In fact, these are always accompanied by 'narrow gorges' in which minima are surrounded by regions of large gradient [97]. Hence, improvement in the trainability of quantum circuits can be achieved by means of 'pre-training' an ansatz such that its initial parameter values are sufficiently close to the optimal solution. As such, initialization by means of pre-training has received significant attention lately [21–31], and are proposed as one of the most promising methods to requantize certain VQAs [32].

In this section, we demonstrate the use of $\mathfrak{g}$-sim for the initialization of VQAs. The idea (Sec. IV B 1) is to perform pre-training on a related auxiliary problem that induces a scalable Lie algebra, and to transfer the solution found as initial parameters for the circuit addressing the target problem. It is expected that the closer the auxiliary problem is to the target one, the more efficient such a transfer will be. As a first example (Sec. IV B 2), we study ground state preparation of the longitudinal-transverse field Ising model (LTFIM) via solving the transverse field Ising model (TFIM) in the first place. LTFIM only differs from TFIM by a few additional generators (the longitudinal fields) and we find substantial improvement both in terms of the fidelity of the ground states prepared and the magnitude of the initial gradients when utilizing pre-training. More surprisingly, we also show that in problems of QAOA (Sec. IV B 3), for which target and auxiliary tasks differ substantially, improved performances can still be achieved over a significant number of (but not all) problem instances.

1. *Pre-training strategy with* $\mathfrak{g}$*-sim*

Let us consider again here a VQE task where the goal is to prepare the ground state of an Hamiltonian H that can be decomposed as $H = \sum_j c_j h_j$, where c_j are real valued coefficients. We then define the Lie algebra $\mathfrak{g}_{\text{target}} = \langle \{ih_j\}\rangle_{\text{Lie}}$, i.e., the algebra generated by the individual terms in H. The goal is to construct a circuit $U(\boldsymbol{\theta})$ to prepare the ground state of H that is generated by some elements of $\mathfrak{g}_{\text{target}}$. In what follows, we will assume that $\dim(\mathfrak{g}_{\text{target}}) \notin \mathcal{O}(\text{poly}(n))$ (else the full VQE problem could be efficiently simulated with $\mathfrak{g}$-sim), meaning that $U(\boldsymbol{\theta})$ should not be constructed from a generating set of $\mathfrak{g}_{\text{target}}$. Hence, the scheme is as follows:

1. Identify a subset of the operators in $i\mathfrak{g}_{\text{target}}$, denoted as $\{h_k\}_k$ such that their Lie closure $\mathfrak{g}_{\text{aux}} = \langle\{ih_k\}\rangle_{\text{Lie}}$, is a subalgebra $\mathfrak{g}_{\text{aux}} \subset \mathfrak{g}_{\text{target}}$ with $\dim(\mathfrak{g}_{\text{aux}}) \in \mathcal{O}(\text{poly}(n))$. Construct a proxy Hamiltonian $H_{\text{aux}} = \sum_k c_k h_k$. As we will see in the examples below, the choice of c_k is informed by the task at hand.

2. On a classical computer, use $\mathfrak{g}$-sim to solve VQE on H_{aux} using an ansatz generated by terms of $\mathfrak{g}_{\text{aux}}$ and with a number of parameters allowing for overparametrization.

3. Extend the trained ansatz with *new* gates generated by (some) of the terms in $\mathfrak{g}_{\text{target}} \setminus \mathfrak{g}_{\text{aux}}$. These new gates are initialized to the identity.

4. On a quantum computer, solve the VQE for H starting from the ansatz constructed in step 3.

Although presented in the context of VQE, similar strategies could be applied to QML problems.

2. *Pre-training VQE for the LTFIM*

We begin by recalling that the LTFIM is a paradigmatic spin-chain model providing a prototypical example of a quantum phase transition. Its Hamiltonian reads

$$H_{\text{LTFIM}} = h_{xx}\sum_{j=1}^{n-1}\sigma_j^x\sigma_{j+1}^x + h_z\sum_{j=1}^{n}\sigma_j^z + h_x\sum_{j=1}^{n}\sigma_j^x\,, \quad (28)$$

where $h_{xx}, h_z, h_x \in \mathbb{R}$. It can be verified that the algebra $\mathfrak{g}_{\text{target}}$ obtained from the terms in H_{LTFIM} has exponential dimension $\mathfrak{g}_{\text{LTFIM}} \in \Theta(4^n)$ [64]. This renders it intractable in $\mathfrak{g}$-sim, induces a barren plateau for deep Hamiltonian variational circuits [64], and precludes efficient overparametrization [65], thus making application of the VQE to identify the ground state of Eq. (28) highly non-trivial.

Instead, by dropping some terms in $H_{\rm LTFIM}$ we obtain the TFIM Hamiltonian, given by

$$H_{\rm TFIM} = h_{xx}\sum_{j=1}^{n-1}\sigma_j^x\sigma_{j+1}^x + h_z\sum_{j=1}^{n}\sigma_j^z\,. \qquad (29)$$

Notably, taking the Lie closure of the terms in $H_{\rm TFIM}$ we obtain $\mathfrak{g}_{\rm aux}\subseteq\mathfrak{g}_0$. Thus, we have successfully identified a subset of operators in $i\mathfrak{g}_{\rm target}$ leading to a an algebra whose dimension is in $\mathcal{O}(\mathrm{poly}(n))$, making it overparametrizable [65] and classically tractable in $\mathfrak{g}$-sim.

The pre-training strategy is now applied to ground state preparations of $H_{\rm LTFIM}$. Following Sec. IV B 1, we begin by constructing an ansatz with $L=\dim(\mathfrak{g}_0)$ and gates generated by terms appearing in $H_{\rm TFIM}$:

$$U(\boldsymbol{\theta}) = \prod_{l=1}^{\dim(\mathfrak{g}_0)} e^{-i\theta_{l,1}\sum_{j=1}^{n-1}\sigma_j^x\sigma_{j+1}^x}e^{-i\theta_{l,2}\sum_{j=1}^{n}\sigma_j^z}, \qquad (30)$$

Parameters of the ansatz are initialized with random uniform values, and we train them to prepare the ground state of $H_{\rm TFIM}$ using $\mathfrak{g}$-sim. Note that since the dynamical Lie algebra $\mathfrak{g}_{\rm aux}$ associated with Eq. (30) is a subalgebra of $\mathfrak{g}_0$, we can use the representation elements that we have pre-computed for $\mathfrak{g}_0$. This can be seen by noting that $e^{-i\theta_{l,1}\sum_{j=1}^{n-1}\sigma_j^x\sigma_{j+1}^x}=\prod_{j=1}^{n-1}e^{-i\theta_{l,1}\sigma_j^x\sigma_{j+1}^x}$, and that each gate generator is an element $\sigma_j^x\sigma_{j+1}^x$ of $\mathfrak{g}_0$. We detail further how to best make use of $\mathfrak{g}$-sim with generators that are sums of Pauli operators in Appendix B 2.

Once the parameters in Eq. (30) are trained to prepare the ground state of $H_{\rm TFIM}$, we modify the circuit by inserting new gates generated by the remaining term of $H_{\rm LTFIM}$, yielding the ansatz

$$U(\boldsymbol{\theta},\boldsymbol{\phi}) = \prod_{l=1}^{\dim(\mathfrak{g}_0)}\Big[e^{-i\theta_{l,1}\sum_{j=1}^{n-1}\sigma_j^x\sigma_{j+1}^x}e^{-i\theta_{l,2}\sum_{j=1}^{n}\sigma_j^z} e^{-i\phi_l\sum_{j=1}^{n}\sigma_j^x}\Big]. \qquad (31)$$

The new gates are initialized to the identity by setting $\boldsymbol{\phi}=(0,0,\ldots)$, while the other gates retain their pre-trained values. Finally, we proceed by training the full ansatz of Eq. (31) to minimize the expectation value of $H_{\rm LTFIM}$. Since $\dim(\mathfrak{g}_{\rm LTFIM})\in\Theta(4^n)$, this last step must be performed using state vector simulations (we use TensorFlow Quantum [98]) thus limiting the system sizes that can be probed.

In Fig. 5, we report a comparison of the pre-training strategy discussed versus uniform random initialization of the circuit parameters, for Hamiltonians with $h_z=-1$ and $h_{xx}=1$. As can be seen in Figures 5(a, b), pre-training yields improvements by several orders of magnitude in terms of the error

$$\epsilon_{\rm LTFIM}(\boldsymbol{\theta},\boldsymbol{\phi}) = \frac{\langle\mathbf{0}|\,U^\dagger(\boldsymbol{\theta},\boldsymbol{\phi})H_{\rm LTFIM}U(\boldsymbol{\theta},\boldsymbol{\phi})\,|\mathbf{0}\rangle - E_{\min}}{\|H_{\rm LTFIM}\|_{\rm HS_1}}, \qquad (32)$$

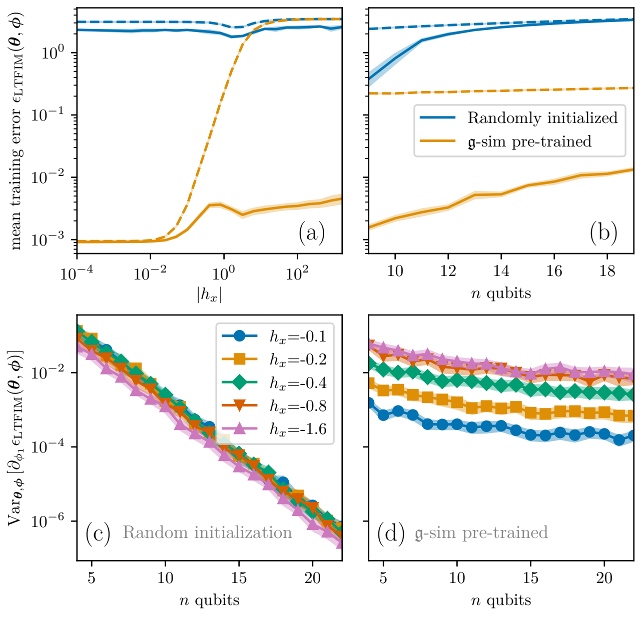


FIG. 5. **Pre-training VQE for LTFIM using $\mathfrak{g}$-sim. (a, b)** Mean training error $\epsilon_{\rm LTFIM}(\boldsymbol{\theta},\boldsymbol{\phi})$ (Eq. (32)) for VQE with random initialization (blue) and pre-trained with $\mathfrak{g}$-sim (orange), (a) at $n=12$ qubits and varying longitudinal field strengths h_x and (b) at field strength $h_x=-1.0$ and varying system sizes n. Dashed lines represent initial ansatz configurations, while solid lines represent trained ansätze. **(c, d)** Comparison of gradient variances at varying system size n and longitudinal field strengths h_x for (c) uniform random parameter initialization and (d) $\mathfrak{g}$-sim pre-training. Shaded bars represent bootstrapped 95% confidence intervals.

where $E_{\min}$ is the ground state energy of $H_{\rm LTFIM}$ obtained by exact diagonalization.

In Fig. 5(a), we compare these errors as a function of the longitudinal field strength $|h_x|$. The prepared state resulting from the pre-training strategy (with errors depicted as an orange dashed line) becomes a poorer approximation to the ground state as $|h_x|$ increases, and even sometimes at par with random initialization (blue dashed line). Still, after the final step of optimization (with errors depicted as plain lines) we found pre-training to consistently enable accurate ground state preparation of $H_{\rm LTFIM}$. Across all the values of h_x studied, the pre-training strategy is the most favorable initialization.

In Fig. 5(b), we report final errors scaling with the system size n, noting that the randomly initialized circuits quickly become untrainable as n increases while the pre-trained ansatz only exhibits mild decline in trainability.

In addition to these improved errors, we observe a mitigation of the barren plateau effect. Due to the exponential dimension of $\mathfrak{g}_{\rm LTFIM}$, in the case of random initialization one would expect gradient variance scaling as [64]

$$\mathrm{Var}_{\boldsymbol{\theta},\boldsymbol{\phi}}\left[\partial_{\phi_m}\epsilon_{\rm LTFIM}(\boldsymbol{\theta},\boldsymbol{\phi})\right]\in\mathcal{O}\left(\frac{1}{2^n}\right), \qquad (33)$$

thus necessitating $\Theta(2^n)$ circuit repetitions to distinguish small gradient values from statistical shot noise. This is verified numerically in Fig. 5(c) which, over varied values of the field h_x, showcases exponentially vanishing gradients. However, in the case of $\mathfrak{g}$-sim pre-training, we can see in Fig. 5(d), that the gradient variances vanish at a much slower rate in n, effectively mitigating appearance of the barren plateau effect and thus extending the scalability of VQE on this system.

3. *Pre-training QAOA*

The quantum approximate optimization algorithm (QAOA) is a VQA that attempts to solve combinatorial optimization problems [99]. Specifically, we consider here its use to approximate solutions of MaxCut problems. We recall that given a graph G with edges E and vertices V, the maximum cut (MaxCut) problem is to find a partition of V into two sets S and T that maximizes the number of edges $e \in E$ having endpoints in both S and T. Encoding a partition as a bitstring $\boldsymbol{z} \in \{0,1\}^n$, its fitness for the MaxCut problem can be quantified by the approximation ratio r defined as

$$r(\boldsymbol{z}) \equiv \frac{C(\boldsymbol{z})}{\arg\max_{\boldsymbol{z}} C(\boldsymbol{z})}, \; C(\boldsymbol{z}) \equiv \sum_{(m,l)\in E} z_m(1-z_l)\,. \quad (34)$$

For general graph problems, finding an exact MaxCut solution is NP-hard [100]. Still, the Goemans-Williamson (GW) algorithm [101] allows one to efficiently find an approximation with a ratio of at least $r_{\mathrm{GW}} \approx 0.878$. To obtain quantum advantage with QAOA, one must outperform this threshold.

QAOA recasts a MaxCut problem as a VQE problem, where the goal is to find the ground state of the phase Hamiltonian

$$H_G = \frac{1}{2}\sum_{(m,l)\in E}(\sigma_m^z\sigma_l^z - I)\,, \quad (35)$$

that depends on the underlying graph G. To prepare such a ground state, one applies a circuit

$$U(\boldsymbol{\beta},\boldsymbol{\gamma}) = \prod_{m=1}^{p} e^{-i\beta_m H_M} e^{-i\gamma_m H_G}\,, \quad (36)$$

where $H_M = \sum_{j=1}^n \sigma_j^x$ is the so-called mixing Hamiltonian, to an initial state $|+\rangle^{\otimes n}$. By optimizing the variational parameters ($\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$) to minimize the expectation value $\langle +|^{\otimes n} U^\dagger(\boldsymbol{\beta},\boldsymbol{\gamma}) H_P U(\boldsymbol{\beta},\boldsymbol{\gamma})|+\rangle^{\otimes n}$ and measuring the resulting state in the computational basis, one may construct approximate solutions to MaxCut. In correspondence to Eq. (34), the approximation ratio of the solution generated by QAOA is defined as

$$r(\boldsymbol{\beta},\boldsymbol{\gamma}) \equiv \frac{\langle +|^{\otimes n} U^\dagger(\boldsymbol{\beta},\boldsymbol{\gamma}) H_G U(\boldsymbol{\beta},\boldsymbol{\gamma})\,|+\rangle^{\otimes n}}{\langle \psi_{\mathrm{GS}}|\, H_G\,|\psi_{\mathrm{GS}}\rangle}\,, \quad (37)$$

where $|\psi_{\mathrm{GS}}\rangle$ is the ground state of the Hamiltonian H_G.

While optimal parameters can be identified for $p = 1$ [102], optimizing them for larger depth (where improved solutions can be found) remains a challenge. Indeed, several works show that general problem instances are likely to experience unfavorable optimization landscapes in the absence of any special underlying structure [103, 104]. These point toward the necessity of finding pre-training strategies for deep-circuit QAOA.

It has been reported that for most choices of G the Lie algebra associated with Eq. (36) will have dimension in $\Omega(2^n)$ [64, 78]. Hence, to apply our pre-training strategy, we need to identify an algebra with polynomial dimension that is related to the original problem. For the circuit of Eq. (36), this is the case for the path graph $G = P_n$ on n-vertices. For such a choice, the Lie algebra $\mathfrak{g}_{\mathrm{QAOA},P_n} = \mathrm{span}\langle\{iH_{P_n}, iH_M\}\rangle_{\mathrm{Lie}}$ is such that (up to a change of basis, where the Pauli σ^x and σ^z are interchanged) $\mathfrak{g}_{\mathrm{QAOA},P_n} \subset \mathfrak{g}_0$, and therefore has $\dim(\mathfrak{g}_{\mathrm{QAOA},P_n}) \in \mathcal{O}(\mathrm{poly}(n))$ (see Appendix F). Hence, we can again use the representation elements of $\mathfrak{g}_0$, and start by training such circuit for $p = \dim(\mathfrak{g}_{\mathrm{QAOA},P_n})$.

The pre-trained parameters $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ can then be used to initialize the modified ansatz

$$U_G(\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\gamma}) = \prod_{l=1}^{p}\left[e^{-i\beta_l H_M} e^{-i\gamma_l H_{P_n}} e^{-i\alpha_l H_G}\right] \quad (38)$$

for any general graph $G \neq P_n$, with the new parameters initialized to $\boldsymbol{\alpha} = (0,0,\ldots)$. We then train the parameters $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, $\boldsymbol{\gamma}$ to minimize the expectation value of H_G.

We study the benefits of this pre-training at $n = 16$ qubits on random 3-regular graphs and Erdös-Rényi graphs with edge probability $p_E = 0.3$. Results are reported in Fig. 6, showing that pre-training significantly outperforms randomly-initialized QAOA circuits.

In terms of the mean approximation ratio r (Fig. 6(a)), both randomly-initialized strategies vastly under-perform the GW threshold r_{GW} (horizontal dashed line), while the pre-training strategy achieves comparable average performance. More strikingly, when looking at details of the approximation ratios r of the individual graph problems (Fig. 6(b)), we find a majority of individual graphs achieve an approximation ratio $r > r_{\mathrm{GW}}$, with the rest at par with random initialization.

Overall, the fraction of solutions that improve on the GW threshold (Fig. 6(c)) converges to 72.5% for 3-regular graphs, and 66.5% for $p_E = 0.3$ Erdös-Rényi graphs. On the other hand, no randomly-initialized circuits achieved better than this threshold. This indicates that $\mathfrak{g}$-sim pre-training is advantageous for QAOA on a substantial fraction of random graphs, even for a relatively crude initialization strategy. We note that even in the case of pre-training, convergence to the improved solutions often requires performing a couple of hundreds of optimization steps. This remains challenging in current quantum devices, and even more so when accounting for noise. Nonetheless starting closer to the solution is always a desirable feature, and further improvements could likely be

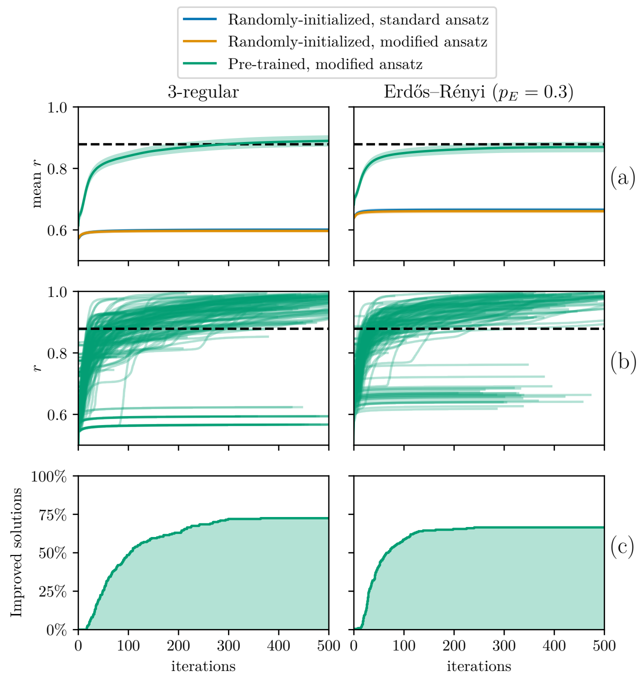


FIG. 6. **Pre-training QAOA using $\mathfrak{g}$-sim.** Performances of QAOA on MaxCut at $n = 16$ qubits for the standard randomly-initialized QAOA ansatz (blue), the modified randomly-initialized ansatz (orange), and the modified ansatz pre-trained with $\mathfrak{g}$-sim (green). Comparisons are made across ensembles of 200 random 3-regular graphs (left panels) and 200 Erdös-Réyni graphs with edge probability $p_E = 0.3$ (right panels). We compare **(a)** the mean approximation ratios r (shaded bars are bootstrapped 95% confidence intervals), **(b)** individual training trajectories, and **(c)** the fraction of solutions outperforming the GW threshold (reported as an horizontal dashed line).

achieved by more carefully aligning the path graph P_n along G, adaptively transforming the ansatz and cost, or resorting to another polynomially-size algebra for the pre-training.

C. Circuit synthesis

Until now, we have been concerned with tasks of state-preparation. We now address more difficult problems of *unitary compilation.* Here, we seek to use $\mathfrak{g}$-sim to identify the parameters of an ansatz circuit $U(\boldsymbol{\theta}) \in \mathcal{G}$ of the form Eq. (1) to implement a target unitary $V \in \mathcal{G}$. Despite the fact that both the target and circuit must belong to a unitary Lie group whose associated Lie algebra is of polynomial dimension, such strategy can already enlarge the reach of $\mathfrak{g}$-sim. In particular, previous uses of $\mathfrak{g}$-sim (as reported in Results 1 & 3) were restricted to certain initial states and observables, but our circuit synthesis compilation goes beyond these cases. Now, $\mathfrak{g}$-sim can be used to classically compile a polynomial-gate-count circuit, and then implement it on a quantum computer to evaluate the evolution of *any* observable or state. More generally, the unitary to be synthesized could be part of a larger protocol such that the initial state or observable may not even be known beforehand. Finally, we note that there may be utility in compiling random unitaries corresponding to polynomially sized Lie algebras, as sampling from the output of some of these circuits have strong hardness guarantees that can be used to demonstrate quantum advantage [105].

The scheme proposed is presented in Sec. IV C 1. We probe its convergence properties for random targets in Sec. IV C 2 displaying polynomial optimization effort in most cases. However, as documented in Sec. IV C 3, given that we work in a reduced representation of the algebra, issues of faithfulness can arise and compromise compilation. Still, we provide a strategy to overcome this issue. Resorting to this strategy we demonstrate exact compilations with $\mathfrak{g}$-sim in a task of dynamical evolution in Sec. IV C 4.

1. *Variational compilation of unitaries in* $\mathfrak{g}$-sim

In variational circuit compilation, we typically seek to train the parameters of an ansatz circuit $U(\boldsymbol{\theta})$ to approximate some target unitary V. Existing techniques for compiling variational circuits to target unitaries usually seek to minimize the cost

$$\mathcal{L}_{\text{HST}}(U(\boldsymbol{\theta}), V) \equiv 1 - \frac{1}{d^2} |\operatorname{Tr}(U^\dagger(\boldsymbol{\theta})V)|^2, \tag{39}$$

that has computational complexity growing quadratically with $d = 2^n$ rendering it quickly classically intractable, and thus would require evaluation on a quantum computer for even modest system sizes. In this situation, evaluation of Eq. (39) could rely on the Hilbert-Schmidt test [106, 107] or on estimation via state sampling [108, 109]. In any case, these assume implementation of the target V in the first place, which is often unrealistic.

In contrast, provided a description of $V \in \mathcal{G}$ with dimension of the associated Lie algebra $\dim(\mathfrak{g}) \in \mathcal{O}(\text{poly}(n))$, the present approach to compilation can be performed entirely on a classical computer with $\mathcal{O}(\text{poly}(n))$ resources. To that intent, we propose the loss function

$$\begin{aligned}\mathcal{L}_{\mathfrak{g}}(U(\boldsymbol{\theta}), V) &\equiv \frac{1}{2\dim(\mathfrak{g})} \|\bar{\boldsymbol{U}}(\boldsymbol{\theta}) - \bar{\boldsymbol{V}}\|_{\text{HS}}^2 \\ &= 1 - \frac{1}{\dim(\mathfrak{g})} \operatorname{Re}\left[\operatorname{Tr}\left(\bar{\boldsymbol{U}}^T(\boldsymbol{\theta})\bar{\boldsymbol{V}}\right)\right], \end{aligned}\tag{40}$$

where $\|\cdot\|_{\text{HS}}$ is again the Hilbert-Schmidt norm and $\bar{\boldsymbol{U}}(\boldsymbol{\theta}) \equiv \Phi_\lambda^{\text{Ad}}(U(\boldsymbol{\theta}))$, $\bar{\boldsymbol{V}} \equiv \Phi_\lambda^{\text{Ad}}(V)$ are the adjoint representations of $U(\boldsymbol{\theta})$ and V, respectively. Gradients of $\mathcal{L}_{\mathfrak{g}}$ can be calculated efficiently, with implementation details provided in Appendix C 2.

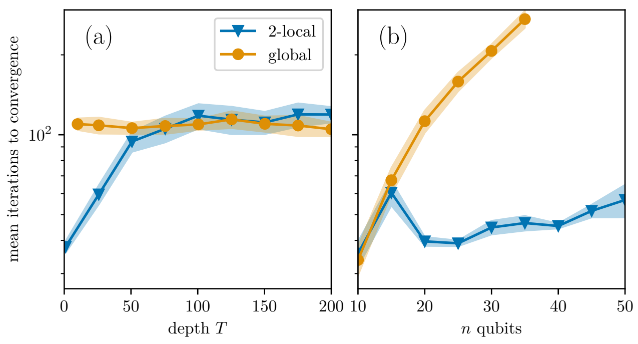


FIG. 7. **Scaling properties of circuit compilation with $\mathfrak{g}$-sim.** We compare the mean number of iterations to converge to a loss (defined in Eq. (40)) $\mathcal{L}_\mathfrak{g} < 10^{-3}$ for random target unitaries (defined in Eq. (41)) with varied levels of locality, including 2-local and global generators. Results for **(a)** varying duration T at $n = 20$ qubits, and for **(b)** varying system size n at duration $T = 10$. The Shaded bars depict bootstrapped 95% confidence intervals. The scaling of the iterations number is found to be polynomial in n and T.

In effect, $\mathcal{L}_\mathfrak{g}$ measures how accurately $U(\boldsymbol{\theta})$ approximates the evolution of the expectation values $\langle G_\alpha \rangle$ under evolution by V for all initial states. Although the Hilbert-Schmidt loss $\mathcal{L}_{\text{HST}}$, in Eq. (39), and its adjoint-space version $\mathcal{L}_\mathfrak{g}$, in Eq. (40), are similar in form, we highlight that $\mathcal{L}_\mathfrak{g}$ is sensitive to global phase differences between $\bar{\boldsymbol{U}}(\boldsymbol{\theta})$ and $\bar{\boldsymbol{V}}$, whereas $\mathcal{L}_{\text{HST}}$ is not sensitive to global phase differences between $U(\boldsymbol{\theta})$ and V. This is because global phase differences in the full Hilbert space are nonphysical, whereas global phase differences in the adjoint representation correspond to a sign flip on all basis observables, which is indeed physical. As we shall see (Sec. IV C 3) related subtleties may compromise our ability to perform faithful compilation. For now we leave these aside, and assess convergence of optimizations with the loss in Eq. (40) for random target unitaries.

2. *Compiling random unitaries*

We first benchmark our compilation scheme against random unitaries in $\mathcal{G}_0 = e^{\mathfrak{g}_0}$, which is the most general and demanding task for this scheme. Any unitary in $\mathcal{G}$ may be written in the form

$$V = e^{-iT \sum_\alpha (\boldsymbol{w})_\alpha G_\alpha}\,, \tag{41}$$

where we fix $|\boldsymbol{w}|_2 = 1$ such that T parametrizes the effective duration of the corresponding dynamics. To generate the weights we sample a matrix from a Haar distribution over the orthogonal group and set the weight vector $\boldsymbol{w}$ equal to one of its columns (potentially padding with zeros to account for elements of the basis not present in the sum).

In general, Eq. (41) involves highly non-local interactions, since many elements of $\mathfrak{g}_0$ are non-local Pauli operators (see Eq. (23)). By excluding greater-than-k-local G_α from Eq. (41), we can refine our study to k-local Hamiltonian dynamics.

We test our scheme by compiling global and 2-local dynamics, both with the 2-local ansatz of Fig. 3(a), (see Appendix E for more details), using $L = \lceil 2 \dim(\mathfrak{g})/K \rceil$ layers, with K the number of generators, ensuring overparametrization. In Fig. 7, we show that the compilation of random unitaries $V \in \mathcal{G}_0$ performs and scales well. At fixed $n = 20$ qubits (Fig. 7(a)), the convergence requirements (measured as the number of optimization iterations required for convergence) plateau to a constant value irrespective of T, indicating that our approach allows fixed-circuit-depth Hamiltonian simulation for arbitrary evolution time. We note that this phenomenon was demonstrated in Ref. [110] on related systems, but with different methods and only targeting local dynamics.

At fixed T (Fig. 7(b)), the number of steps to converge appears constant in n for 2-local targets, and polynomial in n for global ones. The case of global targets with small duration T is detailed further in Appendix G 1. Overall, this demonstrates that our methods can efficiently minimize the loss $\mathcal{L}_\mathfrak{g}$ on an overparametrized ansatz for a range of random unitaries in $\mathcal{G}_0$ with varying locality, system size n, and dynamics duration T, providing a strong foundation for our compilation scheme.

3. *Faithfulness of compilation*

Although we saw consistent convergence with respect to $\mathcal{L}_\mathfrak{g}$ (Fig. 7), one should question whether minimizing $\mathcal{L}_\mathfrak{g}$ is sufficient for unitary compilation. It can be seen from Eq. (40) that $\mathcal{L}_\mathfrak{g}$ is a faithful loss function for unitary training iff Φ_λ^{Ad} is a faithful representation.

As discussed in Sec. II B, faithful representation $\Phi_\mathfrak{g}^{\text{ad}}$ of the Lie algebra does not guarantee faithful representation Φ_λ^{Ad} of the Lie group. In particular, we have already seen in Eq. (12) that for unitaries $W \in Z(\mathcal{G})$ (i.e., for unitaries in the center of the group), it is the case that $\Phi_\lambda^{\text{Ad}}(W) = I$ such that $\Phi_\lambda^{\text{Ad}}(WV) = \Phi_\lambda^{\text{Ad}}(V)$. That is, in the adjoint representation one cannot distinguish V from WV. For the case of $\mathfrak{g}_0$, the center is $Z(\mathcal{G}) = \{(\sigma^z)^{\otimes n}, I\}$ up to a global phase (as detailed in Appendix G 2).

Such an issue can indeed be seen in our numerics. In the first row of Fig. 8(a) we report systematic success in minimizing $\mathcal{L}_\mathfrak{g}$. However, as can be distinguished by the loss $\mathcal{L}_{\text{HST}}$, only half of the optimizations yields the correct target V while the other half rather yields the unitary $(\sigma^z)^{\otimes n} V$ (second and third row). To guarantee successful compilation of V, one must either ensure convergence to the manifold of correct solutions, or be able to flag when an error has occurred such that it could be corrected (i.e., by applying an additional unitary $(\sigma^z)^{\otimes n}$). We now detail a strategy achieving the former.

Any unitary target $V \in \mathcal{G}$ may be written in the form $V = e^{-iTH}$ with $iH \in \mathfrak{g}$ as in Eq. (41). Rather than directly aiming for the compilation of V, we consider a family of intermediary targets $V_m = e^{-iHt_m}$

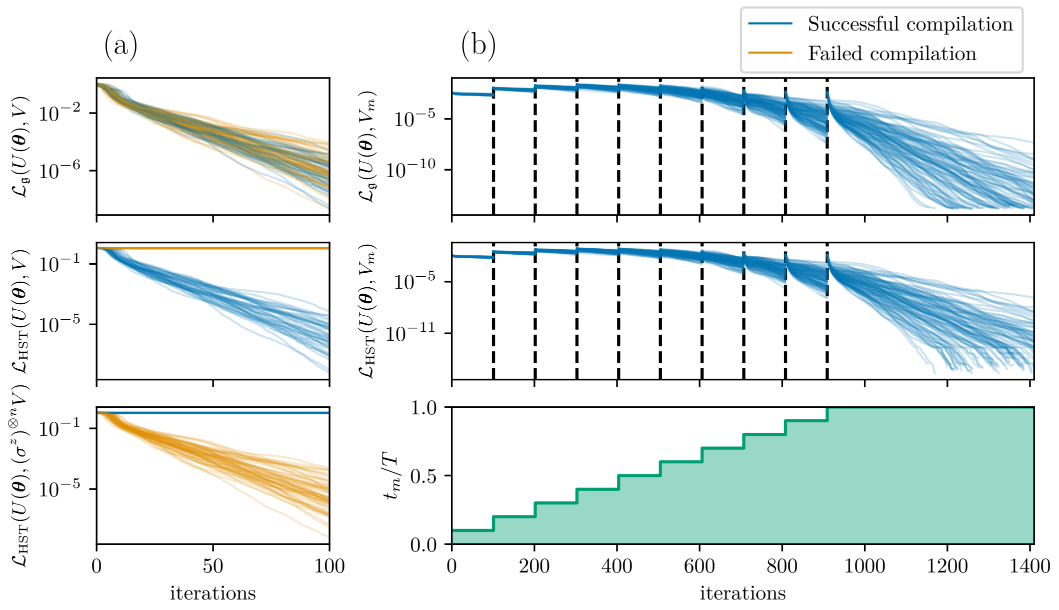


FIG. 8. **Successfully compiling circuits in an algebra with non-trivial group center elements. (a)** Although the classical loss function $\mathcal{L}_{\mathfrak{g}}(U(\boldsymbol{\theta}), V)$ can be efficiently minimized (Fig. 7), this is a necessary but insufficient condition to minimize the Hilbert-Schmidt loss function $\mathcal{L}_{\mathrm{HST}}$, and thus to faithfully compile a target unitary V. We train a linear-depth ansatz $U(\boldsymbol{\theta})$ (see Fig. 3(a)) initialized with uniform random parameters to minimize $\mathcal{L}_{\mathfrak{g}}(U(\boldsymbol{\theta}), V)$ at $T = 1$ and $n = 10$. Although $\mathcal{L}_{\mathfrak{g}}$ is successfully minimized in all instances (top panel), $\mathcal{L}_{\mathrm{HST}}$ is only minimized in approximately half the instances, and rather *maximized* otherwise (middle panel). In all cases where the compilation fails, the optimized ansatz instead approximates $(\sigma^z)^{\otimes n}V$ (bottom panel). **(b)** The scheme outlined in Sec. IV C 3 successfully rectifies this issue. The ansatz is initialized at $\boldsymbol{\theta} = \mathbf{0}$, and compiled to a sequence of intermediate targets V_m corresponding to increased t_m from 0 to T (bottom panel), with the parameters obtained from compiling a target used to initialize the next . For sufficiently small time steps Δt_m, successfully minimizing $\mathcal{L}_{\mathfrak{g}}$ (top panel) ensures minimization of $\mathcal{L}_{\mathrm{HST}}$ (middle panel) and thus faithful compilation of the final target V.

for increasing steps $t_m \in [0, T]$, with $t_0 = 0$. We then solve the corresponding compilation problems $\boldsymbol{\theta}_m = \arg\min_{\boldsymbol{\theta}} \mathcal{L}_{\mathfrak{g}}(U(\boldsymbol{\theta}), V_m)$ sequentially, with $\boldsymbol{\theta}$ initialized to $\boldsymbol{\theta}_{m-1}$ at each step. For sufficiently small $\Delta t_m = t_m - t_{m-1}$, it is expected that no jump between solution manifolds will occur. Crucially, given that $V_0 = I$, we can ensure that we start in the correct manifold by setting $\boldsymbol{\theta}_0 = \mathbf{0}$ such that $U(\boldsymbol{\theta}_0) = I$ (rather than $(\sigma^z)^{\otimes n}$) .

Viability of the proposed scheme is confirmed numerically and reported in Fig. 8(b). For all the random unitaries V assessed, the optimization concludes with parameters replicating accurately the desired target, as evidenced by the low values of $\mathcal{L}_{\mathrm{HST}}(U(\boldsymbol{\theta}), V) < 10^{-6}$. This demonstrates that faithful compilation is possible even when the cost function $\mathcal{L}_{\mathfrak{g}}$ is not faithful.

4. Application to dynamical simulation

As noted in Sec. IV C 2, given a Hamiltonian supported by a Lie algebra $\mathfrak{g}$ with $\dim(\mathfrak{g}) \in \mathcal{O}(\mathrm{poly}(n))$, our scheme enables compilation in polynomial-depth circuits of the corresponding time-evolution operators. This allows the study of dynamics of observables *not* supported by $\mathfrak{g}$-sim and arbitrary states, which are in general not classically tractable. Here we demonstrate the utility of our scheme by synthesizing circuits for the Hamiltonian time-evolution of the TFXY spin chain of Eq. (24) with open boundary conditions and random local magnetic fields b_j drawn from $N(0, \xi^2)$. This Hamiltonian supports the phenomenon of Anderson localization [111], and has previously been utilized to demonstrate related compilation techniques based on Cartan decomposition [110].

We aim to train an ansatz $U(\boldsymbol{\theta}_{t_m})$ to approximate $V_{t_m} \equiv e^{-it_m H_{\mathrm{TFXY}}}$ at a range of discrete times $t_m \in [0, 200]$. For each t_m, the ansatz has a structure

$$U(\boldsymbol{\theta}) = \prod_{l=1}^{L} \Bigg[\prod_{j=1}^{n-1} e^{-i\theta_{l,(n+2j+1)}\sigma_j^x\sigma_{j+1}^x} e^{-i\theta_{l,(n+2j)}\sigma_j^y\sigma_{j+1}^y} \prod_{j=1}^{n} e^{-i\theta_{l,j}\sum_{j=1}^{n}\sigma_j^z} \Bigg], \tag{42}$$

One can verify that the dynamical Lie algebra associated with this circuit is again $\mathfrak{g}_0$ as defined in Eq. (23), meaning that we can again utilize the representation elements already computed. At this point we note that while $U(\boldsymbol{\theta})$ has the exact same structure as that of a first-order Trotterization of any of the V_{t_m} unitaries, each term of the Hamiltonian is associated with an independent trainable parameter in $U(\boldsymbol{\theta})$. This fact is important as $U(\boldsymbol{\theta})$ is not trying to learn a Trotterized version of V_{t_m}. In fact,

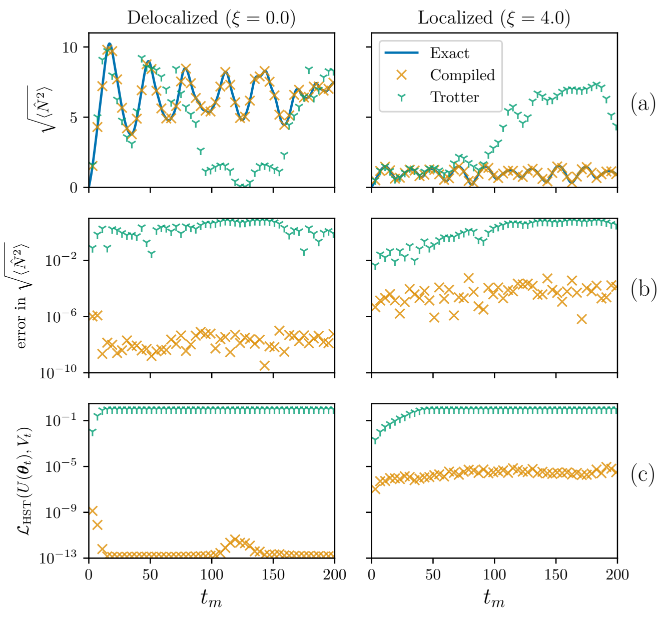


FIG. 9. **Compiling Hamiltonian dynamics with $\mathfrak{g}$-sim.** Dynamics of a single-spin-flip excitation in a TFXY spin chain (Eq. (24) with $n = 12$) in the absence of magnetic fields ($\xi = 0$, left panels) and with random fields inducing Anderson localization ($\xi = 4$, right panels). The exact dynamics (blue) is compared to the dynamics yielded by our compilation scheme (orange X), and first-order Trotterization (green Y). The compilation scheme accurately showcases Anderson localization, reproducing the RMS position $\sqrt{\hat{N}^2}$ of the excitation (a) with low errors (b), while Trotterization quickly diverges beyond small simulation times. Furthermore, comparison of the compiled unitaries to the exact time-evolution operators by means of the Hilbert-Schmidt test $\mathcal{L}_{\mathrm{HST}}$, show small errors at all times (c), guaranteeing faithful dynamics of all observables.

because we use $\mathfrak{g}$-sim, we do not need to ever perform a Trotterization of the target unitary, as V_{t_m} can be compiled exactly for all evolution times t_m. This is due to the fact that one can efficiently compute the adjoint representation $\Phi_\lambda^{\mathrm{Ad}}(V_{t_m}) = e^{-it_m \Phi_{\mathfrak{g}}^{\mathrm{ad}}(H_{\mathrm{TFXY}})}$ on a classical computer. This is advantageous compared to variational compilation schemes for time evolution that necessitate the target to be implementable on a quantum device in the first place, which is often achieved by means of a Trotter approximation [109, 112–115], and therefore introduces approximation errors.

For the sake of concreteness, we focus on the dynamics of a single-spin-flip initial state $|{\downarrow\uparrow\uparrow\uparrow\uparrow\uparrow\uparrow\uparrow\uparrow\uparrow\uparrow\uparrow}\rangle$ for $n = 12$ qubits. In the absence of a magnetic field ($b_j = 0$ for all j) this excitation diffuses throughout the system, but a disordered field ($\xi > 0$) restricts this diffusion (Anderson localization). Following the example of Ref. [110], we study the position operator for the excitation

$$\hat{N} = \sum_{j=1}^{n} (j-1)\frac{1-\sigma_j^z}{2}\,, \tag{43}$$

whose moments provide a key signature of Anderson localization. In particular, for this system the p-th moment $\langle |\hat{N}|^p \rangle$ admits a time-independent upper bound [110, 116]. Note that $\hat{N}^2$ is expressible as a linear combination of elements in the algebra (σ_j^z) and product of elements in the algebra $(\sigma_j^z \sigma_{j'}^z)$. Moreover, given that the algebra is composed of Pauli operators, and since the initial state is separable, we can readily compute the entries of the vector $\boldsymbol{e}^{(\mathrm{in})}$ and the matrix $\boldsymbol{E}^{(\mathrm{in})}$. Hence, the $p = 2$ moment can be efficiently computed with $\mathfrak{g}$-sim. As noted above, larger moments quickly become intractable. Furthermore, we stress that our choice of a classically-tractable system size ($n = 12$ qubits) is purely to enable us to compute the loss $\mathcal{L}_{\mathrm{HST}}(U(\boldsymbol{\theta}_{t_m}), V_{t_m})$ for verification of correct unitary compilation. In general, our scheme can achieve compilation at larger system sizes (e.g., $n = 50$ qubits in Fig. 7).

We compile the time-evolution operators for the case of no magnetic field ($\xi = 0$) and of a disordered magnetic field ($\xi = 4$) with a Hamiltonian renormalized as $H_{\mathrm{TFXY}} \to \frac{H_{\mathrm{TFXY}}}{\|H_{\mathrm{TFXY}}\|_{\mathrm{HS}_1}}$ to eliminate any norm dependence of the dynamics. Our ansatz is determined by Eq. (42) and is composed of $L = 17$ layers, which is overparametrized to ensure trainability. In line with the strategy previously discussed, the circuit is first initialized with $\boldsymbol{\theta}_{t_0} = \boldsymbol{0}$, and thereafter initialized with $\boldsymbol{\theta}_{t_m}$ according to the parameters found in the t_{m-1} step of optimization.

In Fig. 9, we report a comparison between the compiled dynamics and a first-order Trotterization at the same depth (i.e., identical circuit structure with $\theta_{lk} = t/L$). We find that, despite using identical circuit structures, our scheme outperforms Trotterization by several orders of magnitude. Looking at the dynamics of the position operator $\sqrt{\hat{N}^2}$ (Fig. 9(a)) one can see that Trotterization fails to reproduce results beyond $t \approx 60$ while our scheme continues to track them accurately at any of the times considered, with errors smaller than 10^{-6} (Fig. 9(b)). More generally, we find that the compiled unitaries reflect accurately the true time-evolution with $\mathcal{L}_{\mathrm{HST}}(U(\boldsymbol{\theta}_t), V_t) < 10^{-5}$ (Fig. 9(c)), thus guaranteeing that the compiled circuits can faithfully reproduce the dynamics of *all* observables, not just those supported by $\mathfrak{g}$ and products thereof. Furthermore, the errors entailed by the compiled circuit do not appear to vary substantially in the duration of the dynamics. Overall, this shows that for time-evolution operators whose associated Lie algebra $\mathfrak{g}$ has $\dim(\mathfrak{g}) \in \mathcal{O}(\mathrm{poly}(n))$, our $\mathfrak{g}$-sim compilation schemes can determine a much more efficient circuit implementation than standard Trotterization.

D. Supervised quantum machine learning

As a final demonstration, we employ $\mathfrak{g}$-sim for the training of a quantum-phase classifier, showcasing its applicability in the context of QML. Despite being trained fully classically on tractable states, such a classifier could then be employed to classify unknown quantum states. Provided that meaningful training data points and quantum models can be found in algebras with polynomial dimension, such schemes could find utility in real experiments. In fact, this scenario of QML on quantum data is very similar to quantum metrology: protocols can be developed through classical simulations and still have merit when implemented through quantum technology. Additionally, in the spirit of Sec. IV B, our $\mathfrak{g}$-sim approach could form the basis of approximate quantum models that are then refined on a quantum computer.

1. *Supervised QML*

In general problems of supervised QML one assumes repeated access to a training dataset $\mathcal{S} = \{(\rho_s, y_s)\}_{s=1}^N$ consisting of of N pairs of states ρ_s together with labels $y_s = F(\rho_s)$ that have been assigned by an unknown underlying function F. The task is to learn parameters $\boldsymbol{\theta}$ of a function $h_{\boldsymbol{\theta}}$ aiming at approximating F as accurately as possible. Upon successful training, it is then possible to accurately predict the labels of previously unseen states.

As typical in tasks of QML, we consider a model $h_{\boldsymbol{\theta}}$ that relies on the expectation value of an observable O after application of a circuit $U(\boldsymbol{\theta})$. That is, on $\ell_{\boldsymbol{\theta}}(\rho_s) = \mathrm{Tr}\big[OU(\boldsymbol{\theta})\rho_s U^\dagger(\boldsymbol{\theta})\big]$. Training relies upon minimization of a mean-squared error loss function, defined here as

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{N}\sum_{s=1}^{N}(y_s - \ell_{\boldsymbol{\theta}}(\rho_s))^2 \,. \tag{44}$$

2. *Training a binary quantum-phase classifier*

For our numerical study, we apply the $\mathfrak{g}$-sim framework to the classification of ground states across a phase transition of the TFIM, in Eq. (29), at system size $n = 50$ qubits. We consider parameters $h_z, h_{xx} \in [0, 1]$ in order to focus on a single phase transition (binary classification). For $h_z/h_{xx} > 1$, the ground state is in the disordered phase, while for $h_z/h_{xx} < 1$ it is in the antiferromagnetic phase. Furthermore, to ensure that the problem is non-trivial, we 'disguise' the TFIM according to a random $V \in \mathcal{G}$ (Eq. (41) with $T = 10$) such that

$$H_{\text{disguised}} = V H_{\text{TFIM}} V^\dagger. \tag{45}$$

To generate the training and test datasets, we start by randomly sampling values of h_z and h_{xx} and assign labels $y_s = -1$ $(+1)$ to Hamiltonian parameters corresponding to the disordered (antiferromagnetic) phase. Each set of parameters corresponds to a distinct instance of H_{TFIM} and, for each of them, we take the corresponding state ρ_s to be the (approximate) ground state of this Hamiltonian instance. A $\mathfrak{g}$-sim classical representation of ρ_s consists of the vector of expectation values $(\boldsymbol{e}^{(s)})_\alpha = \mathrm{Tr}[\rho_s G_\alpha]$.

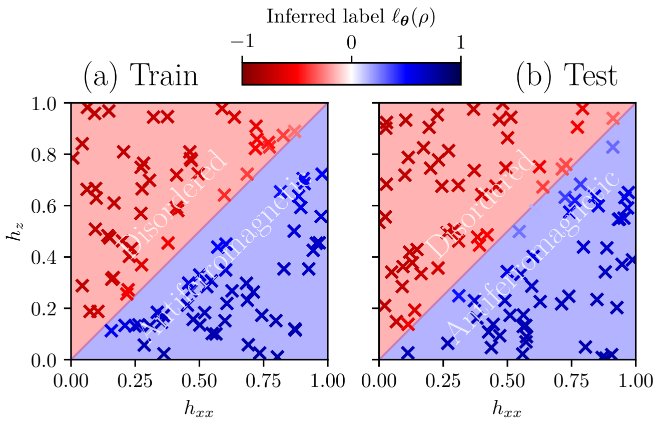


FIG. 10. **Supervised QML with a classically-efficient circuit.** **(a)** Label assignments from a 50-qubit classifier on a training dataset of disguised TFIM ground states for $T = 10$ (details in the main text), across the disordered-antiferromagnetic phase transition. The classifier is trained to minimize the loss function in Eq. (44). **(b)** Inference on a new set of data drawn from the same distribution. The model achieves 100% classification accuracy on the test dataset.

Since exact diagonalization is intractable for $n = 50$ qubits, to compute these, we resort to VQE in $\mathfrak{g}$-sim with an overparametrized ansatz to obtain a circuit $U(\boldsymbol{\theta})$ such that $\rho_s \approx U(\boldsymbol{\theta})\left|0\right\rangle\left\langle 0\right|^{\otimes n} U^\dagger(\boldsymbol{\theta})$. We then compute the classical representation $\boldsymbol{e}^{(s)}$ by applying Eq. (15) to a classical representation of the computational zero state. Finally, we transform this classical representation of the ground state of H_{TFIM} to the corresponding 'disguised' ground state of $H_{\text{disguised}}$ by applying V. Such procedure is repeated to generate a training dataset of 200 labeled states, equally divided in the two phases. To evaluate the performances of the classifier, we apply the same procedure to create a test dataset with the same number of states but that have not been seen by the optimizer during training.

The quantum classifier is realized as 21 layers of the 2-local ansatz in Eq. (42) (see also Appendix E) and a measurement observable $O = \sigma_1^z \in i\mathfrak{g}_0$. We train it to minimize the loss function in Eq. (44) over the training dataset. Once the parameters trained, we assess accuracy of the classifier on the test dataset. Given trained parameters $\boldsymbol{\theta}^*$, to perform inference we assign labels as

$$h_{\boldsymbol{\theta}^*}(\rho_s) = \mathrm{sgn}(\ell_{\boldsymbol{\theta}^*}(\rho_s)) \in \mathcal{Y}\,. \tag{46}$$

Results are depicted in Fig. 10(a), showing that for this problem the classifier achieves 100% classification accuracy on the new data. Consistent classification accuracy is also verified across multiple random instances of the unitary disguise V, thus demonstrating successful application of $\mathfrak{g}$-sim to a supervised QML problem.

We note that in Appendix H, we discuss how the $\mathfrak{g}$-sim phase classifier can be efficiently implemented on a quantum device.

V. CONCLUSIONS

Efficient classical simulations of quantum circuits are a valuable tool in scaling towards quantum advantage. In this work, we have presented $\mathfrak{g}$-sim, a classical simulation and optimization framework that relies on the Lie-algebraic structure of the circuits. Reformulating existing results on Lie-algebraic simulation [43, 44] into a modern presentation aimed at the quantum computing community, we further extended the scope of such simulations to a broader set of initial states, observables, or noise, and improved on the efficiency of their implementations. Moreover by comparing the scope of applicability of $\mathfrak{g}$-sim to that of Wick-based simulations, we argue that our proposed methods enable new regimes for classical simulability that would be otherwise intractable. Of course, such classical simulations are only possible in restricted situations. Conditions allowing for such scalability with $\mathfrak{g}$-sim were laid out, and demonstrations, with system sizes of up to $n = 200$ qubits, highlighting distinctions compared to other classical simulation techniques were provided.

By introducing circuit optimization to this framework, we enabled the scalable classical study of paradigmatic examples relevant for variational quantum computing, demonstrating utility in studying scaling behaviors of VQA problems, improving trainability and mitigating barren plateaus via classical pre-training, and implementing supervised QML problems. These results expand the growing insights in classical pre-training of VQAs [21–31], Lie-algebraic study of trainability [64, 65], and the usefulness of classically simulable variational quantum algorithms [32–42].

Furthermore, we expanded the framework of $\mathfrak{g}$-sim to include compilation and circuit synthesis, which is a novel direction for simulation schemes of this type. By constructing and optimizing a circuit fidelity cost function that can be computed entirely in $\mathfrak{g}$-sim, we demonstrated that one can synthesize linear-depth circuits for unitary transformations in $e^{\mathfrak{g}_0}$, where $\mathfrak{g}_0$ (22) is the algebra used for simulations throughout this work. Compiling such rotations has already proven to be of great utility to the community, with e.g., the use of Givens rotation decomposition to prepare Hartree-Fock states with linear circuit depth [117, 118]. These can be used, e.g., as initial states for more refined circuit preparation [119]. However, our scheme does not necessarily require an underlying free-fermion structure, and therefore expands the class of compilable transformations to include *any* system corresponding to a poly(n)-dimensional algebra.

To date, Lie-algebraic considerations have been pivotal in many topics of quantum science including quantum error correction [120, 121], controllability of quantum systems [62, 63, 122–124], efficient measurements [125], dynamical simulations [47, 126], and studying trainability properties of parametrized quantum circuits [64, 65]. Our work has demonstrated further utility in existing applications (dynamical simulations and studying trainability properties), and expanded this list to include classical pre-training of VQAs, efficient circuit compilation, and supervised QML. We anticipate that the $\mathfrak{g}$-sim framework will provide helpful new perspectives and tools in the development of variational quantum computing.

A. Future work

In this work, we have explored a variety of applications of $\mathfrak{g}$-sim. Nonetheless, several applications beyond the scope of this work are apparent.

One of such applications is quantum error mitigation (QEM) [127], a class of techniques which seek to minimize noise-induced biases in quantum algorithm outputs without resorting to full-fledged quantum error correction. In particular, learning-based QEM methods [16–19] require access to pairs of noisy circuit outputs (obtained from quantum hardware) and corresponding noiseless outputs (obtained from classically-efficient simulation techniques) in order to learn a function mapping noisy outputs to their correct noiseless values. In this context, $\mathfrak{g}$-sim could extend current classical simulation techniques that have been employed.

In a similar vein, $\mathfrak{g}$-sim has potential applications in the context of randomized benchmarking [128]. The greatest limitation in quantum computing is the effect of hardware errors in computations, both coherent and incoherent. There is thus a pressing need to comprehensively characterize the type and magnitude of errors present on quantum hardware. One approach to this is randomized benchmarking, which compares outputs of random gate sequences of increasing length to known expected results without resorting to standard process tomography. Recent work [129] has investigated the use of matchgate circuits for randomized benchmarking. Matchgates are closely related to $\mathfrak{g}_0$, and the underlying simulation schemes have much in common with $\mathfrak{g}$-sim. Much like the case of error mitigation, $\mathfrak{g}$-sim could be used to generate benchmarking data for non-matchgate circuits, expanding the scope of these techniques.

We remark that $\mathfrak{g}$-sim has already played a central role in the study of classical simulability of certain variational quantum algorithms [32–42]. Here, it has been noted that while $\mathfrak{g}$-sim can emulate the information processing capabilities of parametrized quantum circuits, its use still requires a quantum computer to obtain the components of the input state onto the irrep. The latter could necessitates a quantum computer for an initial data-acquisition phase, thus indicating that the whole algorithm is not fully classical end-to-end. It remains an open question for future research whether such "quantum-enhanced" simulations [32] can achieve a quantum advantage (if only polynomial), or whether $\mathfrak{g}$-sim or other classical simulation techniques can be used for effective pre-training.

We also note the opportunity to further improve the compilation scheme in Sec. IV C. Although its performance appears competitive with the Cartan decomposition approach of Ref. [110], one disadvantage is that our approach requires re-optimization at each time step, whereas the former requires only one optimization. We believe that this limitation could be overcome by using a variational fast-forwarding ansatz [112], which should work since the diagonalization unitary must necessarily be in $\mathcal{G}$. Furthermore, our scheme could trivially be extended to the compilation of evolution unitaries for time-dependent Hamiltonians. In the case of Hamiltonians with periodic Floquet driving, it should even be possible to construct a fast-forwardable solution, which to the best of our knowledge would not be possible with Cartan decomposition methods.

While the previous discussion was based on noiseless simulations, noisy simulations also have many applications. Understanding the limitations imposed by noise and ways to mitigate them is primordial, and we expect the noisy simulations capabilities through $\mathfrak{g}$-sim to play an important role in such research. Already, several of the studies can readily be extended to a noisy setup. These include the study of over-parametrization of Sec. IV A that could be performed under Pauli noise channels providing further insights into noise effects when optimizing quantum circuits [130, 131]. The pre-initializion of Sec. IV B and the QML model of Sec. IV D could be re-trained including such noise, leading to more robust circuits.

More generally tasks of state preparation could benefit from the inclusion of noise, with in particular the preparation of Hartree-Fock states [117, 118] that could be optimized to reflect more realistic conditions. Going further, one could envision noise-aware compilation, whereby one would aim at compiling a target unitary such that, depending on details of the noise, different circuit realizations would be identified through optimization. Adequacy of the scalable cost function used for the noiseless case (40) will need to be assessed for this noisy scenario. We leave this aspect for future works. These are all straightforward examples of use of noisy simulations with $\mathfrak{g}$-sim, but more applications could be expected.

As discussed earlier and developed further in the Appendices, several generalizations of $\mathfrak{g}$-sim are possible. In particular, while presented here in the context of digital quantum computations, the methodologies can be easily adapted to analog computing. This opens up the possibility of exploring similar applications in this domain, and we anticipate that Lie-algebraic simulations will be valuable for further investigations of the capabilities and constraints of analog and emerging digital-analog quantum platforms [132–137].

Finally, we recall that by studying the limitations of $\mathfrak{g}$-sim and Wick-based techniques, we uncovered a deep connection between these classical simulation methods and quantum resources theories. Importantly, our results hint at the existence of different, non-equivalent, types of resource which make each of those methods fails. The exploration of this line of thought could lead to further understanding of what makes a quantum process truly "quantum", and thus deepen our understanding on the limitations of classical simulation strategies.

ACKNOWLEDGEMENTS

The authors wish to thank Adrian Chapman, Nahuel L. Diaz, Tyson Jones, Bálint Koczor and Arthur Rattew for helpful technical conversations, and further thank Adrian Chapman for comments on the manuscript. MLG acknowledges the Rhodes Trust for the support of a Rhodes Scholarship. MLG was also supported by the U.S. DOE through a quantum computing program sponsored by the Los Alamos National Laboratory (LANL) Information Science & Technology Institute. ML acknowledges support by the Center for Nonlinear Studies at LANL. ML and MC were supported by the Laboratory Directed Research and Development (LDRD) program of LANL under project numbers 20230049DR and 20230527ECR. LC was supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research through the Accelerated Research in Quantum Computing Program MACH-Q Project. FS was supported by the Laboratory Directed Research and Development (LDRD) program of Los Alamos National Laboratory (LANL) under project number 20220745ER. This work was also supported by LANL's ASC Beyond Moore's Law project and by by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, under Computational Partnerships program. The authors acknowledge the use of the University of Oxford Advanced Research Computing (ARC) facility [138] in carrying out this work.

[1] D. Gottesman, *Stabilizer codes and quantum error correction* (California Institute of Technology, 1997).
[2] G. Vidal, Efficient classical simulation of slightly entangled quantum computations, Phys. Rev. Lett. **91**, 147902 (2003).
[3] G.-C. Wick, The evaluation of the collision matrix, Physical review **80**, 268 (1950).
[4] L. G. Valiant, Quantum computers that can be simulated classically in polynomial time, in *Proceedings of the thirty-third annual ACM symposium on Theory of computing* (2001) pp. 114–123.
[5] B. M. Terhal and D. P. DiVincenzo, Classical simulation of noninteracting-fermion quantum circuits, Physical Review A **65**, 032325 (2002).
[6] Y. Kim, A. Eddins, S. Anand, K. X. Wei, E. Van Den Berg, S. Rosenblatt, H. Nayfeh, Y. Wu, M. Zaletel, K. Temme, *et al.*, Evidence for the utility of quantum computing before fault tolerance, Nature **618**, 500

(2023).

[7] M. Cerezo, A. Arrasmith, R. Babbush, S. C. Benjamin, S. Endo, K. Fujii, J. R. McClean, K. Mitarai, X. Yuan, L. Cincio, and P. J. Coles, Variational quantum algorithms, Nature Reviews Physics **3**, 625–644 (2021).

[8] K. Bharti, A. Cervera-Lierta, T. H. Kyaw, T. Haug, S. Alperin-Lea, A. Anand, M. Degroote, H. Heimonen, J. S. Kottmann, T. Menke, *et al.*, Noisy intermediate-scale quantum algorithms, Reviews of Modern Physics **94**, 015004 (2022).

[9] M. Schuld, I. Sinayskiy, and F. Petruccione, An introduction to quantum machine learning, Contemporary Physics **56**, 172 (2015).

[10] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, Quantum machine learning, Nature **549**, 195 (2017).

[11] M. Larocca, F. Sauvage, F. M. Sbahi, G. Verdon, P. J. Coles, and M. Cerezo, Group-invariant quantum machine learning, PRX Quantum **3**, 030341 (2022).

[12] M. Cerezo, G. Verdon, H.-Y. Huang, L. Cincio, and P. J. Coles, Challenges and opportunities in quantum machine learning, Nature Computational Science 10.1038/s43588-022-00311-3 (2022).

[13] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven, Barren plateaus in quantum neural network training landscapes, Nature Communications **9**, 1 (2018).

[14] M. Cerezo, A. Sone, T. Volkoff, L. Cincio, and P. J. Coles, Cost function dependent barren plateaus in shallow parametrized quantum circuits, Nature Communications **12**, 1 (2021).

[15] E. R. Anschuetz and B. T. Kiani, Beyond barren plateaus: Quantum variational algorithms are swamped with traps, Nature Communications **13**, 7760 (2022).

[16] P. Czarnik, A. Arrasmith, P. J. Coles, and L. Cincio, Error mitigation with Clifford quantum-circuit data, Quantum **5**, 592 (2021).

[17] A. Strikis, D. Qin, Y. Chen, S. C. Benjamin, and Y. Li, Learning-based quantum error mitigation, PRX Quantum **2**, 040330 (2021).

[18] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, A. Bengtsson, S. Boixo, M. Broughton, B. B. Buckley, *et al.*, Observation of separated dynamics of charge and spin in the Fermi-Hubbard model, arXiv preprint arXiv:2010.07965 (2020).

[19] A. Montanaro and S. Stanisic, Error mitigation by training with fermionic linear optics, arXiv preprint arXiv:2102.02120 (2021).

[20] G. Matos, C. N. Self, Z. Papić, K. Meichanetzidis, and H. Dreyer, Characterization of variational quantum algorithms using free fermions, Quantum **7**, 966 (2023).

[21] E. Grant, L. Wossnig, M. Ostaszewski, and M. Benedetti, An initialization strategy for addressing barren plateaus in parametrized quantum circuits, Quantum **3**, 214 (2019).

[22] G. Verdon, M. Broughton, J. R. McClean, K. J. Sung, R. Babbush, Z. Jiang, H. Neven, and M. Mohseni, Learning to learn with quantum neural networks via classical neural networks, arXiv preprint arXiv:1907.05415 (2019).

[23] F. Sauvage, S. Sim, A. A. Kunitsa, W. A. Simon, M. Mauri, and A. Perdomo-Ortiz, Flip: A flexible initializer for arbitrarily-sized parametrized quantum circuits, arXiv preprint arXiv:2103.08572 (2021).

[24] A. Rad, A. Seif, and N. M. Linke, Surviving the barren plateau in variational quantum circuits with Bayesian learning initialization, arXiv preprint arXiv:2203.02464 (2022).

[25] X. Liu, G. Liu, H.-K. Zhang, J. Huang, and X. Wang, Mitigating barren plateaus of variational quantum eigensolvers, IEEE Transactions on Quantum Engineering , 1 (2024).

[26] K. Mitarai, Y. Suzuki, W. Mizukami, Y. O. Nakagawa, and K. Fujii, Quadratic clifford expansion for efficient benchmarking and initialization of variational quantum algorithms, Physical Review Research **4**, 033012 (2022).

[27] G. Ravi, P. Gokhale, Y. Ding, W. Kirby, K. Smith, J. Baker, P. Love, H. Hoffmann, K. Brown, and F. Chong, Cafqa: A classical simulation bootstrap for variational quantum algorithms, arXiv preprint arXiv:2202.12924 (2022).

[28] M. Cheng, K. Khosla, C. Self, M. Lin, B. Li, A. Medina, and M. Kim, Clifford circuit initialisation for variational quantum algorithms, arXiv preprint arXiv:2207.01539 (2022).

[29] J. Dborin, F. Barratt, V. Wimalaweera, L. Wright, and A. G. Green, Matrix product state pre-training for quantum machine learning, Quantum Science and Technology **7**, 035014 (2022).

[30] A. A. Mele, G. B. Mbeng, G. E. Santoro, M. Collura, and P. Torta, Avoiding barren plateaus via transferability of smooth solutions in a Hamiltonian variational ansatz, Physical Review A **106**, L060401 (2022).

[31] M. S. Rudolph, J. Miller, D. Motlagh, J. Chen, A. Acharya, and A. Perdomo-Ortiz, Synergistic pretraining of parametrized quantum circuits via tensor networks, Nature Communications **14**, 8367 (2023).

[32] M. Cerezo, M. Larocca, D. García-Martín, N. L. Diaz, P. Braccia, E. Fontana, M. S. Rudolph, P. Bermejo, A. Ijaz, S. Thanasilp, *et al.*, Does provable absence of barren plateaus imply classical simulability? Or, why we need to rethink variational quantum computing, arXiv preprint arXiv:2312.09121 (2023).

[33] A. Angrisani, Learning unitaries with quantum statistical queries, arXiv preprint arXiv:2310.02254 (2023).

[34] P. Bermejo, P. Braccia, M. S. Rudolph, Z. Holmes, L. Cincio, and M. Cerezo, Quantum convolutional neural networks are (effectively) classically simulable, arXiv preprint arXiv:2408.12739 (2024).

[35] S. Lerch, R. Puig, M. Rudolph, A. Angrisani, T. Jones, M. Cerezo, S. Thanasilp, and Z. Holmes, Efficient quantum-enhanced classical simulation for patches of quantum landscapes, arXiv preprint arXiv:2411.19896 10.48550/arXiv.2411.19896 (2024).

[36] F. J. Schreiber, J. Eisert, and J. J. Meyer, Classical surrogates for quantum learning models, Physical Review Letters **131**, 100803 (2023).

[37] S. Jerbi, C. Gyurik, S. C. Marshall, R. Molteni, and V. Dunjko, Shadows of quantum machine learning, Nature Communications **15**, 5676 (2024).

[38] Y. Shao, F. Wei, S. Cheng, and Z. Liu, Simulating quantum mean values in noisy variational quantum algorithms: A polynomial-scale approach, arXiv preprint arXiv:2306.05804 (2023).

[39] A. Basheer, Y. Feng, C. Ferrie, and S. Li, Alternating layered variational quantum circuits can be classically optimized efficiently using classical shadows, in *Proceedings of the AAAI Conference on Artificial Intelligence*,

Vol. 37 (2023) pp. 6770–6778.

[40] R. Shaffer, L. Kocia, and M. Sarovar, Surrogate-based optimization for variational quantum algorithms, Physical Review A **107**, 032415 (2023).

[41] E. R. Anschuetz, A. Bauer, B. T. Kiani, and S. Lloyd, Efficient classical algorithms for simulating symmetric quantum systems, Quantum **7**, 1189 (2023).

[42] A. A. Mele and Y. Herasymenko, Efficient learning of quantum states prepared with few fermionic non-gaussian gates, arXiv preprint arXiv:2402.18665 (2024).

[43] R. D. Somma, Quantum computation, complexity, and many-body physics, arXiv preprint quant-ph/0512209 (2005).

[44] R. Somma, H. Barnum, G. Ortiz, and E. Knill, Efficient solvability of Hamiltonians and limits on the power of some quantum computational models, Physical Review Letters **97**, 190501 (2006).

[45] L. Schatzki, M. Larocca, Q. T. Nguyen, F. Sauvage, and M. Cerezo, Theoretical guarantees for permutation-equivariant quantum neural networks, npj Quantum Information **10**, 12 (2024).

[46] X. Bonet-Monroig, R. Babbush, and T. E. O'Brien, Nearly optimal measurement scheduling for partial tomography of quantum states, Physical Review X **10**, 031064 (2020).

[47] E. Kökcü, T. Steckmann, Y. Wang, J. Freericks, E. F. Dumitrescu, and A. F. Kemper, Fixed depth hamiltonian simulation via cartan decomposition, Physical Review Letters **129**, 070501 (2022).

[48] S. Qvarfort and I. Pikovski, Solving quantum dynamics with a lie algebra decoupling method, arXiv preprint arXiv:2210.11894 (2022).

[49] A. Barthe, M. Cerezo, A. T. Sornborger, M. Larocca, and D. García-Martín, Gate-based quantum simulation of gaussian bosonic circuits on exponentially many modes, arXiv preprint arXiv:2407.06290 (2024).

[50] R. D. Somma, R. King, R. Kothari, T. O'Brien, and R. Babbush, Shadow hamiltonian simulation, arXiv preprint arXiv:2407.21775 10.48550/arXiv.2407.21775 (2024).

[51] M. T. West, J. Heredge, M. Sevior, and M. Usman, Provably trainable rotationally equivariant quantum machine learning, arXiv preprint arXiv:2311.05873 (2023).

[52] S. Kazi, M. Larocca, M. Farinati, P. J. Coles, M. Cerezo, and R. Zeier, Analyzing the quantum approximate optimization algorithm: ansätze, symmetries, and lie algebras, arXiv preprint arXiv:2410.05187 (2024).

[53] M. Oszmaniec and Z. Zimborás, Universal extensions of restricted classes of quantum operations, Physical review letters **119**, 220502 (2017).

[54] E. Pozzoli, M. Leibscher, M. Sigalotti, U. Boscain, and C. P. Koch, Lie algebra for rotational subsystems of a driven asymmetric top, Journal of Physics A: Mathematical and Theoretical **55**, 215301 (2022).

[55] R. Wiersema, E. Kökcü, A. F. Kemper, and B. N. Bakalov, Classification of dynamical lie algebras of 2-local spin systems on linear, circular and fully connected topologies, npj Quantum Information **10**, 110 (2024).

[56] E. Kökcü, R. Wiersema, A. F. Kemper, and B. N. Bakalov, Classification of dynamical lie algebras generated by spin interactions on undirected graphs, arXiv preprint arXiv:2409.19797 (2024).

[57] G. Aguilar, S. Cichy, J. Eisert, and L. Bittel, Full classification of pauli lie algebras, arXiv preprint arXiv:2408.00081 (2024).

[58] E. Chitambar and G. Gour, Quantum resource theories, Reviews of modern physics **91**, 025001 (2019).

[59] 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, Nature Communications **5**, 1 (2014).

[60] B. C. Hall, *Lie groups, Lie algebras, and representations* (Springer, 2013).

[61] A. Kirillov Jr, *An introduction to Lie groups and Lie algebras*, 113 (Cambridge University Press, 2008).

[62] D. D'Alessandro, *Introduction to Quantum Control and Dynamics*, Chapman & Hall/CRC Applied Mathematics & Nonlinear Science (Taylor & Francis, 2007).

[63] R. Zeier and T. Schulte-Herbrüggen, Symmetry principles in quantum systems theory, Journal of mathematical physics **52**, 113510 (2011).

[64] M. Larocca, P. Czarnik, K. Sharma, G. Muraleedharan, P. J. Coles, and M. Cerezo, Diagnosing Barren Plateaus with Tools from Quantum Optimal Control, Quantum **6**, 824 (2022).

[65] M. Larocca, N. Ju, D. García-Martín, P. J. Coles, and M. Cerezo, Theory of overparametrization in quantum neural networks, Nature Computational Science **3**, 542 (2023).

[66] R. Wiersema, C. Zhou, Y. de Sereville, J. F. Carrasquilla, Y. B. Kim, and H. Yuen, Exploring entanglement and optimization within the Hamiltonian variational ansatz, PRX Quantum **1**, 020319 (2020).

[67] T. Jones and J. Gacon, Efficient calculation of gradients in classical simulations of variational quantum algorithms, arXiv preprint arXiv:2009.02823 (2020).

[68] S. Lloyd, Almost any quantum logic gate is universal, Physical Review Letters **75**, 346 (1995).

[69] R. D. Mattuck, *A guide to Feynman diagrams in the many-body problem* (Courier Corporation, 2012).

[70] R. Gilmore, On the properties of coherent states, Revista Mexicana de Física **23**, 143 (1974).

[71] A. M. Perelomov, Generalized coherent states and some of their applications, Soviet Physics Uspekhi **20**, 703 (1977).

[72] W.-M. Zhang, R. Gilmore, *et al.*, Coherent states: Theory and some applications, Reviews of Modern Physics **62**, 867 (1990).

[73] R. Delbourgo and J. Fox, Maximum weight vectors possess minimal uncertainty, Journal of Physics A: Mathematical and General **10**, L233 (1977).

[74] S. Aaronson and D. Gottesman, Improved simulation of stabilizer circuits, Physical Review A **70**, 052328 (2004).

[75] A. Heimendahl, F. Montealegre-Mora, F. Vallentin, and D. Gross, Stabilizer extent is not multiplicative, Quantum **5**, 400 (2021).

[76] O. Reardon-Smith, The fermionic linear optical extent is multiplicative for 4 qubit parity eigenstates, arXiv preprint arXiv:2407.20934 (2024).

[77] P. Bermejo, P. Braccia, A. A. Mele, N. L. Diaz, A. E. Deneris, M. Larocca, and M. Cerezo, Characterizing quantum resourcefulness via group-fourier decompositions, Manuscript in preparation.

[78] S. Kazi, M. Larocca, M. Farinati, P. J. Coles, M. Cerezo, and R. Zeier, Analyzing the quantum approximate optimization algorithm: ansätze, symmetries, and lie al-

gebras, arXiv preprint arXiv:2410.05187 (2024).
[79] M. L. Goh, g-sim, https://github.com/gohmat/g-sim (2023).
[80] B. Dias and R. Koenig, Classical simulation of non-gaussian fermionic circuits, Quantum **8**, 1350 (2024).
[81] J. Cudby and S. Strelchuk, Gaussian decomposition of magic states for matchgate computations, arXiv preprint arXiv:2307.12654 (2023).
[82] E. Fontana, M. S. Rudolph, R. Duncan, I. Rungger, and C. Cîrstoiu, Classical simulations of noisy variational quantum circuits, arXiv preprint arXiv:2306.05400 (2023).
[83] C. Cirstoiu, A fourier analysis framework for approximate classical simulations of quantum circuits, arXiv preprint arXiv:2410.13856 10.48550/arXiv.2410.13856 (2024).
[84] L. Bittel and M. Kliesch, Training variational quantum algorithms is NP-hard, Phys. Rev. Lett. **127**, 120502 (2021).
[85] B. Neyshabur, Z. Li, S. Bhojanapalli, Y. LeCun, and N. Srebro, The role of over-parametrization in generalization of neural networks, in *International Conference on Learning Representations* (2018).
[86] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, Understanding deep learning (still) requires rethinking generalization, Communications of the ACM **64**, 107 (2021).
[87] Z. Allen-Zhu, Y. Li, and Y. Liang, Learning and generalization in overparameterized neural networks, going beyond two layers, Advances in neural information processing systems (2019).
[88] S. Du, J. Lee, H. Li, L. Wang, and X. Zhai, Gradient descent finds global minima of deep neural networks, in *International Conference on Machine Learning* (PMLR, 2019) pp. 1675–1685.
[89] R.-D. Buhai, Y. Halpern, Y. Kim, A. Risteski, and D. Sontag, Empirical study of the benefits of overparameterization in learning latent variable models, in *International Conference on Machine Learning* (PMLR, 2020) pp. 1211–1219.
[90] S. S. Du, X. Zhai, B. Poczos, and A. Singh, Gradient descent provably optimizes over-parameterized neural networks, in *International Conference on Learning Representations* (2019).
[91] A. Brutzkus, A. Globerson, E. Malach, and S. Shalev-Shwartz, SGD learns over-parameterized networks that provably generalize on linearly separable data, in *International Conference on Learning Representations* (2018).
[92] D. Wecker, M. B. Hastings, and M. Troyer, Progress towards practical quantum variational algorithms, Physical Review A **92**, 042303 (2015).
[93] W. W. Ho and T. H. Hsieh, Efficient variational simulation of non-trivial quantum states, SciPost Phys. **6**, 29 (2019).
[94] C. Cade, L. Mineh, A. Montanaro, and S. Stanisic, Strategies for solving the Fermi-Hubbard model on near-term quantum computers, Physical Review B **102**, 235122 (2020).
[95] M. Cerezo, A. Sone, T. Volkoff, L. Cincio, and P. J. Coles, Cost function dependent barren plateaus in shallow parametrized quantum circuits, Nature Communications **12**, 1 (2021).
[96] Z. Holmes, K. Sharma, M. Cerezo, and P. J. Coles, Connecting ansatz expressibility to gradient magnitudes and barren plateaus, PRX Quantum **3**, 010313 (2022).
[97] A. Arrasmith, Z. Holmes, M. Cerezo, and P. J. Coles, Equivalence of quantum barren plateaus to cost concentration and narrow gorges, Quantum Science and Technology **7**, 045015 (2022).
[98] M. Broughton, G. Verdon, T. McCourt, A. J. Martinez, J. H. Yoo, S. V. Isakov, P. Massey, R. Halavati, M. Yuezhen Niu, A. Zlokapa, E. Peters, O. Lockwood, A. Skolik, S. Jerbi, V. Dunjko, M. Leib, M. Streif, D. Von Dollen, H. Chen, S. Cao, R. Wiersema, H.-Y. Huang, J. R. McClean, R. Babbush, S. Boixo, D. Bacon, A. K. Ho, H. Neven, and M. Mohseni, Tensorflow quantum: A software framework for quantum machine learning, arXiv preprint arXiv:2003.02989 (2020).
[99] E. Farhi, J. Goldstone, and S. Gutmann, A quantum approximate optimization algorithm, arXiv preprint arXiv:1411.4028 (2014).
[100] R. M. Karp, *Reducibility among combinatorial problems* (Springer, 1972).
[101] M. X. Goemans and D. P. Williamson, Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming, Journal of the ACM (JACM) **42**, 1115 (1995).
[102] A. Ozaeta, W. van Dam, and P. L. McMahon, Expectation values from the single-layer quantum approximate optimization algorithm on ising problems, Quantum Science and Technology **7**, 045036 (2022).
[103] L. Zhou, S.-T. Wang, S. Choi, H. Pichler, and M. D. Lukin, Quantum approximate optimization algorithm: Performance, mechanism, and implementation on near-term devices, Physical Review X **10**, 021067 (2020).
[104] G. Koßmann, L. Binkowski, L. van Luijk, T. Ziegler, and R. Schwonnek, Deep-circuit qaoa, arXiv preprint arXiv:2210.12406 (2022).
[105] M. Oszmaniec, N. Dangniam, M. E. Morales, and Z. Zimborás, Fermion sampling: a robust quantum computational advantage scheme using fermionic linear optics and magic input states, PRX Quantum **3**, 020328 (2022).
[106] S. Khatri, R. LaRose, A. Poremba, L. Cincio, A. T. Sornborger, and P. J. Coles, Quantum-assisted quantum compiling, Quantum **3**, 140 (2019).
[107] K. Sharma, S. Khatri, M. Cerezo, and P. J. Coles, Noise resilience of variational quantum compiling, New Journal of Physics **22**, 043006 (2020).
[108] M. C. Caro, H.-Y. Huang, N. Ezzell, J. Gibbs, A. T. Sornborger, L. Cincio, P. J. Coles, and Z. Holmes, Out-of-distribution generalization for learning quantum dynamics, Nature Communications **14**, 3751 (2023).
[109] J. Gibbs, Z. Holmes, M. C. Caro, N. Ezzell, H.-Y. Huang, L. Cincio, A. T. Sornborger, and P. J. Coles, Dynamical simulation via quantum machine learning with provable generalization, Physical Review Research **6**, 013241 (2024).
[110] E. Kökcü, T. Steckmann, Y. Wang, J. Freericks, E. F. Dumitrescu, and A. F. Kemper, Fixed depth hamiltonian simulation via cartan decomposition, Physical Review Letters **129**, 070501 (2022).
[111] P. W. Anderson, Absence of diffusion in certain random lattices, Physical review **109**, 1492 (1958).
[112] C. Cirstoiu, Z. Holmes, J. Iosue, L. Cincio, P. J. Coles, and A. Sornborger, Variational fast forwarding for quan-

tum simulation beyond the coherence time, npj Quantum Information **6**, 1 (2020).
[113] S.-H. Lin, R. Dilip, A. G. Green, A. Smith, and F. Pollmann, Real-and imaginary-time evolution with compressed quantum circuits, PRX Quantum **2**, 010342 (2021).
[114] N. F. Berthusen, T. V. Trevisan, T. Iadecola, and P. P. Orth, Quantum dynamics simulations beyond the coherence time on noisy intermediate-scale quantum hardware by variational trotter compression, Phys. Rev. Res. **4**, 023097 (2022).
[115] M. L. Goh and B. Koczor, Direct estimation of the density of states for fermionic systems, arXiv preprint arXiv:2407.03414 (2024).
[116] V. Bucaj, On the kunz-souillard approach to localization for the discrete one dimensional generalized anderson model, arXiv preprint arXiv:1608.01379 (2016).
[117] I. D. Kivlichan, J. McClean, N. Wiebe, C. Gidney, A. Aspuru-Guzik, G. K.-L. Chan, and R. Babbush, Quantum simulation of electronic structure with linear depth and connectivity, Physical review letters **120**, 110501 (2018).
[118] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, S. Boixo, M. Broughton, B. B. Buckley, D. A. Buell, *et al.*, Hartree-fock on a superconducting qubit quantum computer, Science **369**, 1084 (2020).
[119] W. Mizukami, K. Mitarai, Y. O. Nakagawa, T. Yamamoto, T. Yan, and Y.-y. Ohnishi, Orbital optimized unitary coupled cluster theory for quantum computer, Physical Review Research **2**, 033421 (2020).
[120] P. Zanardi and M. Rasetti, Noiseless quantum codes, Physical Review Letters **79**, 3306 (1997).
[121] B. Eastin and E. Knill, Restrictions on transversal encoded quantum gate sets, Physical review letters **102**, 110502 (2009).
[122] N. Khaneja, R. Brockett, and S. J. Glaser, Time optimal control in spin systems, Physical Review A **63**, 032308 (2001).
[123] Z. Zimborás, R. Zeier, T. Schulte-Herbrüggen, and D. Burgarth, Symmetry criteria for quantum simulability of effective interactions, Physical Review A **92**, 042309 (2015).
[124] I. Marvian, Restrictions on realizable unitary operations imposed by symmetry and locality, Nature Physics **18**, 283 (2022).
[125] T.-C. Yen and A. F. Izmaylov, Cartan subalgebra approach to efficient measurements of quantum observables, PRX Quantum **2**, 040320 (2021).
[126] T. Steckmann, T. Keen, A. F. Kemper, E. F. Dumitrescu, and Y. Wang, Simulating the mott transition on a noisy digital quantum computer via cartan-based fast-forwarding circuits, arXiv preprint arXiv:2112.05688 (2021).
[127] Z. Cai, R. Babbush, S. C. Benjamin, S. Endo, W. J. Huggins, Y. Li, J. R. McClean, and T. E. O'Brien, Quantum error mitigation, Reviews of Modern Physics **95**, 045005 (2023).
[128] E. Knill, D. Leibfried, R. Reichle, J. Britton, R. B. Blakestad, J. D. Jost, C. Langer, R. Ozeri, S. Seidelin, and D. J. Wineland, Randomized benchmarking of quantum gates, Physical Review A **77**, 012307 (2008).
[129] J. Helsen, S. Nezami, M. Reagor, and M. Walter, Matchgate benchmarking: Scalable benchmarking of a continuous family of many-qubit gates, Quantum **6**, 657 (2022).
[130] D. García-Martín, M. Larocca, and M. Cerezo, Effects of noise on the overparametrization of quantum neural networks, Phys. Rev. Res. **6**, 013295 (2024).
[131] M. Duschenes, J. Carrasquilla, and R. Laflamme, Characterization of overparametrization in the simulation of realistic quantum systems, Physical Review A **109**, 062607 (2024).
[132] L. Lamata, A. Parra-Rodriguez, M. Sanz, and E. Solano, Digital-analog quantum simulations with superconducting circuits, Advances in Physics: X **3**, 1457981 (2018).
[133] A. Parra-Rodriguez, P. Lougovski, L. Lamata, E. Solano, and M. Sanz, Digital-analog quantum computation, Physical Review A **101**, 022305 (2020).
[134] O. R. Meitei, B. T. Gard, G. S. Barron, D. P. Pappas, S. E. Economou, E. Barnes, and N. J. Mayhall, Gate-free state preparation for fast variational quantum eigensolver simulations: ctrl-vqe, arXiv preprint arXiv:2008.04302 (2020).
[135] L. Henriet, L. Beguin, A. Signoles, T. Lahaye, A. Browaeys, G.-O. Reymond, and C. Jurczak, Quantum computing with neutral atoms, Quantum **4**, 327 (2020).
[136] A. J. Daley, I. Bloch, C. Kokail, S. Flannigan, N. Pearson, M. Troyer, and P. Zoller, Practical quantum advantage in quantum simulation, Nature **607**, 667 (2022).
[137] J. Wurtz, A. Bylinskii, B. Braverman, J. Amato-Grill, S. H. Cantu, F. Huber, A. Lukin, F. Liu, P. Weinberg, J. Long, *et al.*, Aquila: Quera's 256-qubit neutral-atom quantum computer, arXiv preprint arXiv:2306.11727 (2023).
[138] A. Richards, University of oxford advanced research computing, DOI: 10.5281/zenodo.22558 (2015).
[139] J. Côté, F. Sauvage, M. Larocca, M. Jonsson, L. Cincio, and T. Albash, Diabatic quantum annealing for the frustrated ring model, arXiv preprint arXiv:2212.02624 (2022).
[140] S. Machnes, E. Assémat, D. Tannor, and F. K. Wilhelm, Tunable, flexible, and efficient optimization of control pulses for practical qubits, Physical review letters **120**, 150401 (2018).
[141] R. T. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud, Neural ordinary differential equations, Advances in neural information processing systems **31** (2018).
[142] F. Schäfer, M. Kloc, C. Bruder, and N. Lörch, A differentiable programming method for quantum control, Machine Learning: Science and Technology **1**, 035009 (2020).
[143] F. Sauvage and F. Mintert, Optimal control of families of quantum gates, Physical Review Letters **129**, 050507 (2022).
[144] L. M. Robledo, Sign of the overlap of hartree-fock-bogoliubov wave functions, Physical Review C—Nuclear Physics **79**, 021302 (2009).
[145] H.-Y. Huang, R. Kueng, and J. Preskill, Predicting many properties of a quantum system from very few measurements, Nature Physics **16**, 1050 (2020).

APPENDICES FOR "LIE-ALGEBRAIC CLASSICAL SIMULATIONS FOR VARIATIONAL QUANTUM COMPUTING"

Here we present additional details for the main results in our manuscript. In Appendix A, we present proofs supporting the ability to perform noiseless and noisy simulations through $\mathfrak{g}$-sim. In Appendix B, we describe several techniques that significantly improve the efficiency of our implementation of $\mathfrak{g}$-sim. Further expanding these improvements, in Appendix C we present efficient methods for computing gradients to enable optimization of circuits and dynamics with respect to observable cost functions. Then, Appendix E contains the ansatz circuits used throughout the work and further details on the benchmarking procedure used to evaluate the performance of our $\mathfrak{g}$-sim implementation. In Appendix F, we note several algebras of interest closely related to $\mathfrak{g}_0$. In Appendix G, we outline some subtleties of unitary compilation with $\mathfrak{g}$-sim. Finally, in Appendix H we comment on practical implementation of a $\mathfrak{g}$-sim-trained phase classifier on a quantum device.

Appendix A: Theorems and proofs

1. Representations of unitary evolution

Lemma 1 (Invariance of the algebra)**.** *Suppose $U \in \mathcal{G}$ and $iG_\alpha \in \mathfrak{g}$. Then defining $\tilde{G}_\alpha = U^\dagger G_\alpha U$ for any G_α, we have $i\tilde{G}_\alpha \in \mathfrak{g}$. That is, $i\mathfrak{g}$ is an invariant subspace as per Definition 3.*

Proof. Since $U \in \mathcal{G}$, there exists some $iH \in \mathfrak{g}$ such that $U = e^{-iH}$. The exponential mapping is defined by

$$e^{-iH} = 1 - iH + \frac{(iH)^2}{2!} - \frac{(iH)^3}{3!} + \dots\,, \tag{A1}$$

and it follows that

$$\tilde{G}_\alpha = G_\alpha + i\,[H, G_\alpha] - \frac{1}{2}\,[H,[H,G_\alpha]] + \dots\,. \tag{A2}$$

Since all the terms of Eq. (A2) are generated by nested commutators only involving elements of $\mathfrak{g}$ (up to appropriate factors of i for each term), by Definition 1 of the Lie algebra we have that $i\tilde{G}_\alpha \in \mathfrak{g}$. □

Lemma 2 (Adjoint representation of unitary evolution, adapted from Appendix C in Ref. [43])**.** *Suppose that $U \in \mathcal{G}$, and that $\{B_\alpha^{(\lambda)}\}_{\alpha=1}^{\dim(\mathcal{L}_\lambda)}$ forms a Schmidt-orthonormal basis of $\mathcal{L}_\lambda$. Defining $\tilde{B}_\alpha^{(\lambda)} = U^\dagger B_\alpha^{(\lambda)} U$, we have*

$$\tilde{B}_\alpha^{(\lambda)} = \sum_\beta u_{\alpha\beta} B_\alpha^{(\lambda)}\,, \tag{A3}$$

where the corresponding matrix elements $u_{\alpha\beta}$ are those of the adjoint representation of U,

$$u_{\alpha\beta} = \left(\Phi_\lambda^{\mathrm{Ad}}(U)\right)_{\alpha\beta}\,. \tag{A4}$$

Proof. Since $U \in \mathcal{G}$, there exists some $iH \in \mathfrak{g}$ such that $U = e^{-iH}$. Furthermore, U may be decomposed as

$$U = \lim_{M\to\infty} \prod_{m=1}^{M} U^{(\Delta)}, \tag{A5}$$

corresponding to infinitesimal steps $\Delta = 1/M$ and infinitesimal unitaries $U^{(\Delta)}$ defined as

$$U^{(\Delta)} = e^{-i\Delta H} = 1 - i\Delta H + \mathcal{O}(\Delta^2). \tag{A6}$$

Given the definition of the invariant space $\mathcal{L}_\lambda$, and the fact that $U^{(\Delta)} \in \mathcal{G}$ we can always write

$$U^{(\Delta)^\dagger} B_\alpha^{(\lambda)} U^{(\Delta)} = B_\alpha^{(\lambda)} - \sum_\beta v_{\alpha\beta}^{(\Delta)} B_\beta^{(\lambda)}. \tag{A7}$$

Using Equations (A6) and (A7) we obtain

$$U^{(\Delta)\dagger} B_\alpha^{(\lambda)} U^{(\Delta)} = B_\alpha^{(\lambda)} + i[\Delta H, B_\alpha^{(\lambda)}] + \mathcal{O}(\Delta^2) = B_\alpha^{(\lambda)} - \sum_\beta v_{\alpha\beta}^{(\Delta)} B_\beta^{(\lambda)}. \tag{A8}$$

We may discard vanishing terms in $\mathcal{O}(\Delta^2)$ since we are in the infinitesimal limit $M \to \infty$. Doing so, and taking $\operatorname{Tr}\Big[B_\gamma^{(\lambda)} \ldots\Big]$ of both sides of Eq. (A8) yields

$$i \operatorname{Tr}\Big[B_\gamma^{(\lambda)} [\Delta H, B_\alpha^{(\lambda)}]\Big] = -\sum_\beta v_{\alpha\beta}^{(\Delta)} \operatorname{Tr}\Big[B_\gamma^{(\lambda)} B_\beta^{(\lambda)}\Big] \tag{A9}$$

$$\implies -\sum_k \Delta h_k \operatorname{Tr}\Big[B_\gamma^{(\lambda)}, [G_k, B_\alpha^{(\lambda)}]\Big] = i\Delta v_{\alpha\gamma}^{(\Delta)} \tag{A10}$$

$$\implies v_{\alpha\gamma}^{(\Delta)} = i\Delta \sum_k h_k (\Phi_{\mathfrak{g}}^{\mathrm{ad}}(G_k))_{\alpha\gamma} = i\Delta \left(\Phi_{\mathfrak{g}}^{\mathrm{ad}}(H)\right)_{\alpha\gamma}, \tag{A11}$$

where we have expanded $H = \sum_k h_k G_k$ for $h_k \in \mathbb{R}$ in terms of a Schmidt-orthonormal basis $\{G_k\}$ of $\mathfrak{g}$ and used the Definition 4 of the adjoint representation. Substituting the result of Eq. (A11) into Eq. (A7) reveals that the infinitesimal group operation is determined entirely by the adjoint representation $\Phi_{\mathfrak{g}}^{\mathrm{ad}}(H)$

$$U^{(\Delta)\dagger} B_\alpha^{(\lambda)} U^{(\Delta)} = \sum_\beta \left(I - i\Delta \Phi_{\mathfrak{g}}^{\mathrm{ad}}(H)\right)_{\alpha\beta} B_\beta^{(\lambda)} = \sum_\beta \left(e^{-i\Delta \Phi_{\mathfrak{g}}^{\mathrm{ad}}(H)}\right)_{\alpha\beta} B_\beta^{(\lambda)} + \mathcal{O}(\Delta^2). \tag{A12}$$

We may again discard terms in $\mathcal{O}(\Delta^2)$ since we are in the infinitesimal limit $M \to \infty$. Noting that the full unitary is obtained by composing infinitesimal steps (Eq. (A5)), we obtain from Eq. (A12) that

$$\tilde{B}_\alpha^{(\lambda)} = U^\dagger B_\alpha^{(\lambda)} U = \sum_\beta \left(e^{-i\Phi_{\mathfrak{g}}^{\mathrm{ad}}(H)}\right)_{\alpha\beta} B_\beta^{(\lambda)} = \sum_\beta \left(\Phi_\lambda^{\mathrm{Ad}}(U)\right)_{\alpha\beta} B_\beta^{(\lambda)}, \tag{A13}$$

which is exactly the desired result. □

2. Evolution of product of observables

Here, we provide details about how product of observables are simulated with $\mathfrak{g}$-sim. When specializing to a product of either a single or two observables, we retrieve the setting of Results. 1 and 3 of the main text, respectively. However, this encapsulates more general situations. In the following, we wish to evaluate the expectation value

$$\langle O \rangle := \operatorname{Tr}\Big[O U \rho^{(\mathrm{in})} U^\dagger\Big], \quad \text{where } O = \prod_a O^{(a)} \quad \text{and } O^{(a)} = \sum_\alpha (\boldsymbol{w}^{(a)})_\alpha B_\alpha^{(\lambda_a)}. \tag{A14}$$

Each $O^{(a)}$ is supported by a single irrep $\mathcal{L}_{\lambda_a}$, and we recall that the unitary realized by the circuit is defined through

$$U = \prod_{l=1}^{L} e^{-i\theta_l H_l}, \tag{A15}$$

with all the generators $iH_l \in \mathfrak{g}$ such that $U \in \mathcal{G}$. For notational brevity, we dropped the dependency on the circuit parameters $\boldsymbol{\theta}$ in our notation.

First note that from the Definition. 5 of the adjoint representation we have $\Phi_\lambda^{\mathrm{Ad}}(VV') = \Phi_\lambda^{\mathrm{Ad}}(V)\Phi_\lambda^{\mathrm{Ad}}(V')$ for any V and $V' \in \mathcal{G}$. Hence, we can obtain the adjoint representation of the overall circuit through

$$\Phi_\lambda^{\mathrm{Ad}}(U) = \left(\prod_{l=1}^{L} e^{-i\theta_l \bar{\boldsymbol{H}}_l}\right). \tag{A16}$$

with $\bar{\boldsymbol{H}}_l \equiv \Phi_\lambda^{\mathrm{ad}}(H_l)$ the adjoint representations of the generators H_l. Eq. (A16) specifies how elements $B_\alpha^{(\lambda)}$ of the irrep basis are transformed under conjugation by the circuit:

$$U^\dagger B_\alpha^{(\lambda)} U = \sum_\beta \left(\Phi_\lambda^{\mathrm{Ad}}(U)\right)_{\alpha\beta} B_\beta^{(\lambda)}. \tag{A17}$$

Second, making use of the cyclicity of the trace and of the identity $UU^\dagger = I$, we get

$$\langle O\rangle = \mathrm{Tr}\left[\left(\prod_a \tilde{O}^{(a)}\right)\rho^{(\mathrm{in})}\right], \quad \text{where } \tilde{O}^{(a)} = U^\dagger O^{(a)} U. \tag{A18}$$

Furthermore, from Eq. (A17), we know that we can write

$$\tilde{O}^{(a)} = \sum_\alpha (\tilde{\boldsymbol{w}}^{(a)})_\alpha B_\alpha^{(\lambda_a)}, \quad \text{with } \tilde{\boldsymbol{w}} = \boldsymbol{w}\cdot\Phi_\lambda^{\mathrm{Ad}}(U). \tag{A19}$$

By combining this expression to Eq. (A18) we are in measure to evaluate the expectation value. In the following we start by the case where O is the product of a single (equivalent to Result. 1) and two observables (equivalent to Result. 3) before addressing to the more general case.

(Product of a single observable) For the case where $O \in \mathcal{L}_\lambda$ we obtain:

$$\langle O\rangle = \sum_\alpha (\tilde{\boldsymbol{w}})_\alpha \,\mathrm{Tr}\left[B_\alpha^{(\lambda)}\rho^{(\mathrm{in})}\right] = \tilde{\boldsymbol{w}}\cdot\boldsymbol{e}, \tag{A20}$$

where we have defined the vector of expectation values $\boldsymbol{e}$ such that $(\boldsymbol{e})_\alpha \equiv \mathrm{Tr}\Big[B_\alpha^{(\lambda)}\rho^{(\mathrm{in})}\Big]$.

(Product of two observables) For $O = O^{(1)}O^{(2)}$ with $O^{(1)} \in \mathcal{L}_{\lambda_1}$ $O^{(1)} \in \mathcal{L}_{\lambda_2}$ we obtain

$$\langle O\rangle = \sum_{\alpha_1,\alpha_2} (\tilde{\boldsymbol{w}}^{(1)})_{\alpha_1}(\tilde{\boldsymbol{w}}^{(2)})_{\alpha_2}\,\mathrm{Tr}\left[B_{\alpha_1}^{(\lambda_1)}B_{\alpha_2}^{(\lambda_2)}\rho^{(\mathrm{in})}\right] = \tilde{\boldsymbol{w}}^{(1)}\cdot\boldsymbol{E}\cdot\tilde{\boldsymbol{w}}^{(2)}. \tag{A21}$$

where we have defined the matrix of expectation values $\boldsymbol{E}$ such that $\boldsymbol{E}_{\alpha\beta} \equiv \mathrm{Tr}\Big[B_\alpha^{(\lambda_1)}B_\beta^{(\lambda_2)}\rho^{(\mathrm{in})}\Big]$.

(Product of observables) For the product of M observables, with $a = 1,\ldots,M$ in Eq. (A14), we get

$$\langle O\rangle = \sum_{\alpha_1,\ldots,\alpha_M}\left(\prod_{a=1}^M \tilde{\boldsymbol{w}}_{\alpha_a}^{(a)}\right)\mathrm{Tr}\left[\left(\prod_{a=1}^M B_{\alpha_a}^{(\lambda_a)}\right)\rho^{(\mathrm{in})}\right] = \sum_{\alpha_1,\ldots,\alpha_M}\boldsymbol{T}_{\alpha_1,\ldots,\alpha_M}\tilde{\boldsymbol{w}}_{\alpha_1}\ldots\tilde{\boldsymbol{w}}_{\alpha_M}, \tag{A22}$$

where we have defined the tensor of expectation values $\boldsymbol{T}$ such that $\boldsymbol{T}_{\alpha_1,\ldots,\alpha_M} \equiv \mathrm{Tr}\Big[\Big(\prod_{a=1}^M B_{\alpha_a}^{(\lambda_a)}\Big)\rho^{(\mathrm{in})}\Big]$.

To conclude, let us briefly comment on the different complexities entailed. We assume that the terms $e^{-i\theta_l\bar{\boldsymbol{H}}_l}$ appearing in Eq. (A16) are known (i.e., that the exponentiation has already been performed) and aim at assessing the complexities entailed when evaluating Eq. (A 2). (i) First, we can evaluate the distinct $\tilde{\boldsymbol{w}}^{(a)}$ through matrix-vector multiplication with vector dimensions $\dim(\mathcal{L}_{\lambda_a})$ for each of the irreps $\mathcal{L}_{\lambda_a}$ involved. (ii) Second, we need to evaluate a number of $\prod_a \dim(\mathcal{L}_{\lambda_a})$ expectation values to obtain the tensor $\boldsymbol{T}$. (iii) Lastly, we need to contract the tensor of expectation values together with the vectors $\tilde{\boldsymbol{w}}^{(a)}$. Overall we see that the complexity scales as $\mathcal{O}(\max_a \dim(\mathcal{L}_{\lambda_a})^M)$. This complexity is dominated by the steps (ii) and (iii) and scales exponentially with the number of observables involved in the product decomposition of O. As mentioned in the main text, the decomposition of an observable to be simulated through $\mathfrak{g}$-sim is not unique and different decompositions can yield different computational scalings.

3. Evolution of observables under noisy channels

In this Appendix, we extend results of the previous section in the presence of noise in the circuit: In Sec. A 3 a, we formalize the noisy setting under consideration and address the case where the observable is supported by a single irrep. In Sec. A 3 b, we rather consider the case of product of observables, each supported in a single irrep, as we shall see this case requires more care.

a. Noisy simulation with observables supported by a single irrep

In the presence of noise, the unitary of Eq. (A15) is now replaced by the channel

$$\tilde{\mathcal{U}} = \bigcirc_{l=1}^L(\tilde{\Lambda}_l \circ \mathcal{U}_l), \quad \text{with } \mathcal{U}_l(\cdot) := U_l\,\cdot\,U_l^\dagger \;\text{ and }\; U_l = e^{-i\theta_l H_l}. \tag{A23}$$

Accordingly, the state obtained after the noisy circuit evolution is now given by $\rho^{(\text{out})} \equiv \tilde{\mathcal{U}}(\rho^{(\text{in})})$. Each of the $\tilde{\Lambda}_l$ terms capture the noise affecting the l-th gate of the circuit and can decomposed as

$$\tilde{\Lambda}_l(\cdot) = \sum_k E_{k,l} \cdot E^\dagger_{k,l}, \tag{A24}$$

in terms of Kraus operators $E_{k,l}$ that satisfy $\sum_k E^\dagger_{k,l} E_{k,l} = I$. When all the $\tilde{\Lambda}_l$ are the identity channel we retrieve the unitary evolution specified in Eq. (A15). To capture the type of noise that can be simulated with $\mathfrak{g}$-sim we recall the following definition from the main text.

Definition 8 (Normalizer of $\mathcal{L}_\lambda$)**.** *Given an irrep $\mathcal{L}_\lambda$, we define its normalizer as all the operators that leaves the subspace invariant under conjugation:*

$$\mathrm{N}(\mathcal{L}_\lambda) := \{A \in \mathcal{L} \,|\, A^\dagger X A \in \mathcal{L}_\lambda, \, \forall X \in \mathcal{L}_\lambda\}. \tag{A25}$$

We highlight that by definition of the irreps we have $\mathcal{G} \subset \mathrm{N}(\mathcal{L}_\lambda)$ for any $\mathcal{L}_\lambda$. However, the normalizer may contain many more operators $A \notin \mathcal{G}$. In analogy to Definition 4, we can represent the action by conjugation of elements of $\mathrm{N}(\mathcal{L}_\lambda)$ onto $\mathcal{L}_\lambda$ in terms of $\dim(\mathcal{L}_\lambda) \times \dim(\mathcal{L}_\lambda)$ matrices:

Definition 9 (Adjoint representation of $\mathrm{N}(\mathcal{L}_\lambda)$)**.** *Given an operator A in $\mathrm{N}(\mathcal{L}_\lambda)$, its adjoint representation on the irrep $\mathcal{L}_\lambda$ is obtained via the map $\Phi^{\mathrm{N}}_\lambda : \mathrm{N}(\mathcal{L}_\lambda) \mapsto \mathbb{R}^{\dim(\mathcal{L}_\lambda)\times\dim(\mathcal{L}_\lambda)}$ and is defined by*

$$\left(\Phi^{\mathrm{N}}_\lambda(A)\right)_{\alpha\beta} \equiv \mathrm{Tr}\left[B^{(\lambda)}_\alpha A^\dagger B^{(\lambda)}_\beta A\right]. \tag{A26}$$

By linearity and Definition 8 we see that, if $E_{k,l} \in \mathrm{N}(\mathcal{L}_\lambda)$ for all k in the Kraus decomposition of Eq. (A24) then

$$\forall X \in \mathcal{L}_\lambda, \quad \tilde{\Lambda}_l(A) \in \mathcal{L}_\lambda. \tag{A27}$$

That is, if a noise channel has a Kraus decomposition with all its Kraus operators belonging to the normalizer, then the action of this channel also leaves $\mathcal{L}_\lambda$ invariant. In such case, we can define the action of the noise channel through

$$\Phi_\lambda(\tilde{\Lambda}_l) \equiv \sum_k \Phi^{\mathrm{N}}_\lambda(E_{k,l}). \tag{A28}$$

One can readily verify that given an observable $O = \sum_\alpha (\boldsymbol{w})_\alpha B^{(\lambda)}_\alpha$, supported by a single irrep $\mathcal{L}_\lambda$, we get that $\tilde{\Lambda}_l(O) = \sum_\alpha (\boldsymbol{w}')_\alpha B^{(\lambda)}_\alpha$ where we can obtain the new weights as $\boldsymbol{w}' = \Phi_\lambda(\tilde{\Lambda}_l) \cdot \boldsymbol{w}$. We are now in measure to generalize Res. 1 to noisy simulations:

Result 4 (Noisy simulation of observables in the λ-th irrep)**.** *Consider a noisy circuit as per Eq. (A23) with noise channels $\tilde{\Lambda}_l$ having a Kraus decomposition as given in Eq. (A24) with operators $E_{k,l} \in \mathrm{N}(\mathcal{L}_\lambda)$ for all k and l. Let O be an observable with support in $\mathcal{L}_\lambda$ such that $O = \sum_\alpha (\boldsymbol{w})_\alpha B^{(\lambda)}_\alpha$ for $\boldsymbol{w} \in \mathbb{R}^{\dim(\mathcal{L}_\lambda)}$. Then, given $\boldsymbol{e}^{(\text{in})}_\lambda$, we can compute*

$$\langle O(\boldsymbol{\theta})\rangle = \boldsymbol{w} \cdot \boldsymbol{e}^{(\text{out})}_\lambda, \tag{A29}$$

with the vector of output expectation values obtained as

$$\boldsymbol{e}^{(\text{out})}_\lambda = \left(\prod_{l=1}^{L} \Phi(\tilde{\Lambda}_l) e^{-i\theta_l \bar{\boldsymbol{H}}_l}\right) \cdot \boldsymbol{e}^{(\text{in})}_\lambda, \tag{A30}$$

where $\bar{\boldsymbol{H}}_l \equiv \Phi^{\mathrm{ad}}_\lambda(H_l)$ and $\Phi(\tilde{\Lambda}_l) \equiv \sum_k \Phi^{\mathrm{N}}_\lambda(E_{k,l})$.

Notably, Result 4 enables us to perform *exact* noisy simulations while retaining the same time and memory complexity as the noiseless case. This is in contrast with typical noisy simulations that either are (i) approximate and based on simulating many pure state trajectories, corresponding to random realization of the noise, or are (ii) exact but incur a quadratic increase in memory and computing requirements due to manipulations of density matrices. We note that the case of noiseless simulations of operators in many irreps (Result 2) is readily ported to the noisy case in a similar fashion, and again do not incur additional overhead. However, as is now discussed the case of product of operators (Result 3) requires more care to be ported to the noisy setup and incurs additional limitations.

b. *Simulation with products of observables each supported by a single irrep*

As per Eq. (A14), we now consider the case where $O = \prod_a O^{(a)}$ and $O^{(a)} \in \mathcal{L}_{\lambda_a}$ for any $a = 1, \dots, M$. The reason why the case with product of observables requires more care is because we cannot deal with each of the observables $O^{(a)}$ separately. In particular, we see that while

$$\mathcal{U}_l(O) = U^\dagger O U = U^\dagger O^{(1)} U U^\dagger O^{(2)} \dots U^\dagger O^{(M)} U = \prod_{a=1}^{M} U^\dagger O^{(a)} U \tag{A31}$$

holds when the channel is unitary, the same is not true for arbitrary noise channel:

$$\tilde{\Lambda}_l^\dagger(O) \neq \prod_{a=1}^{M} \tilde{\Lambda}_l^\dagger(O^{(a)}). \tag{A32}$$

Still, upon further restrictions of the noise channels, we can perform noisy simulation of product of observables with $\mathfrak{g}$-sim. In particular we now require that the noise channels acting on our circuits adopt the form:

$$\tilde{\Lambda}_l(\cdot) = \sum_{k=1}^{K} p_{k,l} V_{k,l} \cdot V_{k,l}^\dagger, \tag{A33}$$

with the $p_{k,l}$ valid distributions for each l (i.e., $p_{k,l} \geqslant 0$ and $\sum_k p_{k,l} = 1$) and where the $V_{k,l} \in \mathrm{N}(\mathcal{L}_{\lambda_a})$ need to be unitaries that all belong to the normalizer of each $\mathcal{L}_{\lambda_a}$. For ease of notations, we have assumed the same number K of unitaries involved for each noise channel.

Eq. (A33) is a restriction of the Kraus decomposition in Eq. (A24), but still encompasses many noise channels of interest such as the Pauli noise channels, whereby each of the $V_{k,l}$ is a Pauli string. Now we have that

$$\tilde{\Lambda}_l^\dagger(O) = \sum_k p_{k,l} \prod_{a=1}^{M} V_{k,l}^\dagger O^{(a)} V_{k,l}, \tag{A34}$$

that resembles Eq. (A31), except for the weighted sum, and can be used for the purpose of simulations. In particular, denoting as $\mathcal{V}_{k,l}(\cdot) := V_{k,l} \cdot V_{k,l}^\dagger$, the expectation value in Eq. (A14) can be written as

$$\langle O \rangle = \sum_{\boldsymbol{k}} p_{\boldsymbol{k}} \operatorname{Tr}\left[\left(\prod_a \tilde{O}_{\boldsymbol{k}}^{(a)}\right) \rho^{(\mathrm{in})}\right], \quad \text{where } \tilde{O}_{\boldsymbol{k}}^{(a)} = \bigcirc_{l=L}^{1} (\mathcal{U}_l^\dagger \circ \mathcal{V}_{\boldsymbol{k}_l,l}^\dagger)(O^{(a)}). \tag{A35}$$

where we have defined $\boldsymbol{k} = \{0, \dots, K\}^L$ a vector of noise trajectories, with entries $\boldsymbol{k}_l$ indicating the unitary $V_{\boldsymbol{k}_l,l}$ that has been applied at layer l, and $p_{\boldsymbol{k}} = \prod_{l=1}^{L} p_{\boldsymbol{k}_l,l}$ the probability associated to the trajectory. Denoting as $\Phi_{\lambda_a}^{\mathrm{N}}(V_{k,l})$ the adjoint representation of the $V_{k,l} \in \mathrm{N}(\mathcal{L}_{\lambda_a})$, we get that

$$\tilde{O}_{\boldsymbol{k}}^{(a)} = \sum_\alpha (\tilde{\boldsymbol{w}}_{\boldsymbol{k}}^{(a)})_\alpha B_\alpha^{(\lambda_a)}, \quad \text{with } \tilde{\boldsymbol{w}}_{\boldsymbol{k}} = \boldsymbol{w} \cdot \prod_{l=1}^{L} \left(\Phi_{\lambda_a}^{\mathrm{N}}(V_{\boldsymbol{k}_l,l}) \Phi_{\lambda_a}^{\mathrm{Ad}}(U_l)\right). \tag{A36}$$

Overall, we can estimate Eq. (A35) through the sampling of S noise trajectories $\boldsymbol{k}$ with probability $p_{\boldsymbol{k}}$, and evaluating the corresponding expectation values through Eq. (A36) and the noiseless results for product of observables detailed in Sec. A 2. The variance of such estimates scales as $1/S$ and the complexity as $\mathcal{O}(S \max_a \dim(\mathcal{L}_{\lambda_a})^M)$ as per Sec. A 2.

4. Analog computing

So far, we have only considered the evolution of states under the action of quantum circuits (or corresponding mixed-unitary channels). Still, the principles underpinning $\mathfrak{g}$-sim can equally be applied to the case where the system is *continuously* driven (i.e., in the analog computing paradigm). In the following, we limit ourselves to the noiseless case where the observable of interest is supported by a single irrep, but stress that extensions to the case of product of observables in differerent irreps (as discussed earlier in Sec. A 2) or to the noisy case (as discussed earlier in Sec. A 3 and with similar restrictions on the noise that can be supported) readily follow.

In the following, we consider evolution under a time-dependent Hamiltonian

$$\mathcal{H}(t) = \sum_h c_h(t) H_h, \tag{A37}$$

with constituents $iH_h \in \mathfrak{g}$ and control functions $c_h(t)$. Given an initial state $\rho^{(in)}$, the state of the system at time t can be expressed as $\rho(t) = U(t)\rho^{(in)}U^\dagger(t)$ where $U(t) = \mathcal{T}\big[\exp(-i\int_0^t \mathcal{H}(t')\,dt')\big]$ with $\mathcal{T}$ the time-ordering operator. In analogy to simulation in the quantum circuit scenario, we are now tasked with evaluating the expectation value $\langle O\rangle \equiv \mathrm{Tr}[O\rho(T)]$ of an observable O supported by $\mathfrak{g}$ for the state $\rho(t=T)$ obtained at the end of the evolution.

Defining (in the Heisenberg picture) the time-dependent operator $O(t) \equiv U^\dagger(t)OU(t)$, we have that $\langle O\rangle = \mathrm{Tr}[O(T)\rho^{(\mathrm{in})}]$. Furthermore, the time evolution of $O(t)$ satisfies the ordinary differential equation (ODE):

$$\frac{d}{dt}O(t) = i[\mathcal{H}(t), O(t)], \tag{A38}$$

with initial condition $O(t=0) = O$. From the Definition 4, it can be verified that

$$[\mathcal{H}(t), B_\alpha^{(\lambda)}] = \sum_\beta \left[\Phi_\lambda^{\mathrm{ad}}(\mathcal{H}(t))\right]_{\alpha\beta} B_\beta^{(\lambda)}, \tag{A39}$$

with the matrix $\Phi_\lambda^{\mathrm{ad}}(\mathcal{H}(t)) = \sum_h c_h(t)\Phi_\lambda^{\mathrm{ad}}(H_h)$.

Hence, decomposing $O(t) = \sum_\alpha (\mathbf{o}(t))_\alpha B_\alpha^{(\lambda)}$ in the observable basis $\{B_\alpha^{(\lambda)}\}$ of the irrep $\mathcal{L}_\lambda$, one can recast Eq. (A38) as the ODE

$$\frac{d}{dt}\mathbf{o}(t) = i\Phi_\lambda^{\mathrm{ad}}(\mathcal{H})\mathbf{o}(t) \tag{A40}$$

over the vector $\mathbf{o}(t)$ that has dimension $\dim(\mathcal{L}_\lambda)$.

In summary, evaluating the expectation value $\langle O\rangle$ consists in two steps. First, solve the dynamics of $\mathbf{o}(t)$ from $t=0$ to $t=T$ according to Eq. (A40). This is achieved by standard numerical ODE solvers, and only requires acting on a $\dim(\mathcal{L}_\lambda)$ vector space. Then, upon evaluation of $\mathbf{o}(T)$, one can compute $\langle O\rangle = \mathbf{o}(T)\cdot \mathbf{e}^{(\mathrm{in})}$, where we recall that the vector $\boldsymbol{e}^{(\mathrm{in})}$ contains the expectation values $\mathrm{Tr}\left[B_\alpha^{(\lambda)}\rho^{(\mathrm{in})}\right]$ of all $B_\alpha^{(\lambda)}$ with respect to the initial state.

Overall, this extends the scope of $\mathfrak{g}$-sim to analog (and digital-analog) quantum computing scenarios. As an example of application, in Ref. [139] such capabilities were used to evaluate (and optimize) continuous diabatic schedules for tasks of ground state preparations over system sizes of up to $n = 39$ spins.

Appendix B: Efficient implementation of $\mathfrak{g}$-sim

The $\mathfrak{g}$-sim framework outlined in Section II C allows scalable classical simulations for products of observables supported by irreps of $\mathcal{G}$ whenever $\dim(\mathfrak{g}) \in \mathcal{O}(\mathrm{poly}(n))$. In addition to extending the scope of previous Lie-algebraic simulation techniques [43, 44], another contribution of this work is to introduce new optimizations which substantially improve the efficiency of $\mathfrak{g}$-sim. In the following subsections, we detail three main improvements:

- Pre-computing eigendecompositions of gate generator adjoint representations reduces the time complexity of $\mathfrak{g}$-sim evolution from $\mathcal{O}(\dim(\mathcal{L}_\lambda)^3)$ to $\mathcal{O}(\dim(\mathcal{L}_\lambda)^2)$ (B 1).

- Relevant matrices are extremely sparse when $\mathcal{L}_\lambda$ has a Pauli basis, allowing speedups by use of sparse methods, even for subalgebras whose generators are not sparse (B 2).

- Eigendecompositions of gates with Pauli generators can be computed in $\mathcal{O}(\dim(\mathcal{L}_\lambda))$ in the representation of a Pauli basis, instead of the usual $\mathcal{O}(\dim(\mathcal{L}_\lambda)^3)$ (B 3).

In practice, we found such optimizations to significantly accelerate our simulations, and all the numerics presented in this work rely on a Python implementation of $\mathfrak{g}$-sim incorporating them.

1. Efficient evolution

Evaluating each of the LK terms $e^{-i\theta\bar{\boldsymbol{H}}_k}$ appearing in Eq. (15) of the simulation routine has time complexity scaling as $\mathcal{O}(\dim(\mathcal{L}_\lambda)^3)$ using standard matrix exponentiation routines. However, given the full-rank eigendecompositions

$$\bar{\boldsymbol{H}}_k = \bar{\boldsymbol{Q}}_k \mathrm{diag}(\epsilon_k^{(1)}, \dots, \epsilon_k^{(\dim(\mathcal{L}_\lambda))})\bar{\boldsymbol{Q}}_k^T \tag{B1}$$

of the (adjoint representation of the) gate generators $\bar{\boldsymbol{H}}_k$, the action of these parametrized gates is given as

$$\Phi_\lambda^{\mathrm{Ad}}(e^{-i\theta H_k}) = \bar{\boldsymbol{Q}}_k \mathrm{diag}(e^{-i\theta\epsilon_k^{(1)}}, \dots, e^{-i\theta\epsilon_k^{(\dim(\mathcal{L}_\lambda))}})\bar{\boldsymbol{Q}}_k^T, \tag{B2}$$

which can be applied to a vector $\boldsymbol{e} \in \mathbb{R}^{\dim(\mathcal{L}_\lambda)}$ with $\mathcal{O}(\dim(\mathcal{L}_\lambda)^2)$ time complexity. Notably, these decompositions only need to be performed once per algebra and per gate generator used, the number of which is typically no more than polynomial in the system size n for relevant applications. Pre-computing these eigendecompositions naïvely scales as $\mathcal{O}(\dim(\mathcal{L}_\lambda)^3)$, however, in Appendix B 3 we detail a procedure by which the eigendecompositions for any Pauli-basis irrep (e.g. $i\mathfrak{g}_0$) can be computed in $\mathcal{O}(\dim(\mathcal{L}_\lambda))$. An explicit algorithm for simulation incorporating these pre-computed eigendecompositions is provided in Algorithm 1.

Algorithm 1 Efficient evolution of observable expectation values using $\mathfrak{g}$-sim.

```
Input: Circuit parameters $\boldsymbol{\theta}$, input observables $\boldsymbol{e}^{(\mathrm{in})}$, circuit generator eigendecompositions (Eq. (B2)) $\bar{\boldsymbol{Q}}_k$ and $\epsilon_k^{(g)}$
  $\boldsymbol{e}^{(\mathrm{out})} \leftarrow \boldsymbol{e}^{(\mathrm{in})}$
  for $l$ in $1, \dots, L$ do
    for $m$ in $1, \dots, M$ do
      $\boldsymbol{e}^{(\mathrm{out})} \leftarrow \bar{\boldsymbol{Q}}_k^T \boldsymbol{e}^{(\mathrm{out})}$
      for $g$ in $1, \dots, \dim(\mathcal{L}_\lambda)$ do
        $(\boldsymbol{e}^{(\mathrm{out})})_g \leftarrow e^{-i\theta_{lk}\epsilon_k^{(g)}}(\boldsymbol{e}^{(\mathrm{out})})_g$
      end for
      $\boldsymbol{e}^{(\mathrm{out})} \leftarrow \bar{\boldsymbol{Q}}_k \boldsymbol{e}^{(\mathrm{out})}$
    end for
  end for
Output: Vector $\boldsymbol{e}^{(\mathrm{out})}$ of basis observables in $\mathcal{L}_\lambda$
Asymptotic complexity: $\mathcal{O}(M \dim(\mathcal{L}_\lambda)^2)$
```

2. Sparsity of adjoint representation

When the relevant basis of $\mathcal{L}_\lambda$ consists of Pauli strings and the gate generators are also Pauli strings (e.g. when performing simulations in $i\mathfrak{g}_0$ where all gates have Pauli operator generators), the adjoint representations $\bar{\boldsymbol{H}}_l$ of the gate generators and their corresponding parametrized gates are of an extremely sparse form, having fewer than $2\dim(\mathcal{L}_\lambda)$ nonzero entries per $(\dim(\mathcal{L}_\lambda) \times \dim(\mathcal{L}_\lambda))$-dimensional matrix. Thus, substantial speedups can be obtained by leveraging sparse linear algebra libraries. This is the case for most simulations presented in this work.

However, we sometimes must apply gates whose generators are not Pauli strings, such as $e^{-i\theta\sum_{j=1}^{n-1}\sigma_j^x\sigma_{j+1}^x}$ for VQE on the LTFIM (Sec. IV B 2), or $e^{-i\theta\sum_{j=1}^{n-1}\sigma_j^z\sigma_{j+1}^z}$ for QAOA (Sec. IV B 3). For gates like these, the adjoint representation of the generator does *not* admit an efficient sparse representation, and will in general be dense. Figure 11 depicts this difference in sparsity.

For gates not generated by single Pauli strings, but rather sums of commuting Pauli strings, one can factorize the gate and still use sparse methods. For example

$$e^{-i\theta\sum_{j=1}^{n-1}\sigma_j^x\sigma_{j+1}^x} = \prod_{j=1}^{n-1} e^{-i\theta\sigma_j^x\sigma_{j+1}^x} \tag{B3}$$

$$\implies \underbrace{\Phi_\lambda^{\mathrm{Ad}}\left(e^{-i\theta\sum_{j=1}^{n-1}\sigma_j^x\sigma_{j+1}^x}\right)}_{\text{dense}} = \prod_{j=1}^{n-1} e^{-i\theta\overbrace{\Phi_\lambda^{\mathrm{ad}}(\sigma_j^x\sigma_{j+1}^x)}^{\text{sparse}}}, \tag{B4}$$

since the adjoint representation is a homomorphism. However, in such cases the dense nature of the eigenvector matrix can make GPU-accelerated dense linear algebra methods highly appealing. For example, we consider tests on

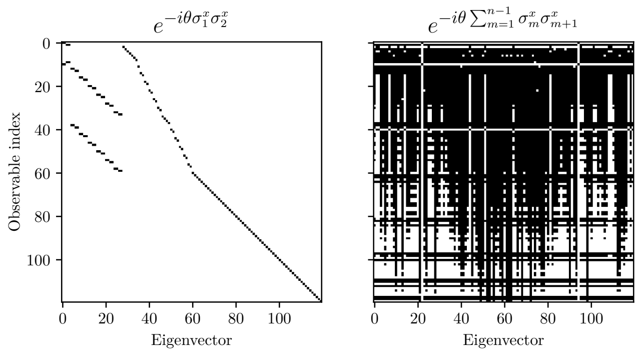


FIG. 11. **Sparsity of gates generated by Pauli strings versus general gates.** We plot sparsity maps for the eigenvector matrix $\bar{\boldsymbol{Q}}$ where $\Phi_\lambda^{\mathrm{Ad}}(\exp(-i\theta H)) = \bar{\boldsymbol{Q}}\mathrm{diag}(e^{-i\theta\epsilon^{(1)}},\ldots,e^{-i\theta\epsilon^{(\dim(\mathfrak{g}))}})\bar{\boldsymbol{Q}}^T$, for $H=\sigma_1^x\sigma_2^x$ (left) and $H=\sum_{m=1}^{n-1}\sigma_m^x\sigma_{m+1}^x$ (right), for a system of $n=8$ qubits in the $\mathcal{L}_\lambda = i\mathfrak{g}_0$ representation. White (black) entries in the map indicates (non) zero entries. The gate $\exp(-i\theta\sigma_1^x\sigma_2^x)$ has a sparse eigenvector matrix since it is generated by a single Pauli string, whereas the gate $\exp\left(-i\theta\sum_{m=1}^n\sigma_m^x\sigma_{m+1}^x\right)$ has a dense eigenvector matrix since it is generated by a sum of Pauli strings.

our hardware (a single core of an Intel Xeon Platinum 8268 for CPU methods vs a NVIDIA Tesla V100 for GPU methods), for our implementation of $\mathfrak{g}$-sim. We found that for systems with sparse gates only, GPU acceleration provided an overall speedup of a factor of 3-5 versus a single CPU core, whereas for systems with dense gates, GPU acceleration provided an overall speedup of a factor of 60-100. The choice of sparse or dense methods and CPU or GPU hardware must be made judiciously depending on problem type and the need for parallelizability.

3. Efficient calculation and eigendecomposition of Pauli-basis adjoint representations

For our Lie-algebraic simulations, we must be able to compute the adjoint representations (Definition 4) of our gate generators H_k:

$$\left(\bar{\boldsymbol{H}}_k\right)_{\alpha\beta} = -\operatorname{Tr}\left[H_k\left[G_\alpha, G_\beta\right]\right]. \tag{B5}$$

naïvely computing an adjoint representation via Eq. (B5) would scale as $\mathcal{O}(\dim(\mathcal{L}_\lambda)^2)$. However, if the basis operators G_α of $\mathcal{L}_\lambda$ are Pauli strings

$$G_\alpha = \bigotimes_{m=1}^{n}\sigma_m^{s_{\alpha,m}}, \quad s_{\alpha,m}\in\{x,y,z\}, \tag{B6}$$

as is the case for $i\mathfrak{g}_0$, and H_k is also a Pauli string, then for fixed k and α (β), Eq. (B5) has a nonzero value for at most one value of β (α). That is, each row and column have at most one nonzero entry. Thus, one can construct the adjoint representation for $\bar{\boldsymbol{H}}_k$ in just $\mathcal{O}(\dim(\mathcal{L}_\lambda))$ by iterating over $\alpha\in\{1,\ldots,\dim(\mathcal{L}_\lambda)\}$, inverting Eq. (B5) to find the Pauli string σ such that $\operatorname{Tr}[H_k[G_\alpha,\sigma]]$ (which is possible since by assumption H_k and G_α are Pauli strings), checking if $\sigma = G_\beta$ for some β (which can be done in $\mathcal{O}(n)$ by using a hashmap), and populating the corresponding element if so.

Similarly, although naïvely computing the eigendecomposition (B1) would be $\mathcal{O}(\dim(\mathcal{L}_\lambda)^3)$, one may compute it in just $\mathcal{O}(\dim(\mathcal{L}_\lambda))$ in the case where the H_k and G_α are Pauli strings. Since the $\bar{\boldsymbol{H}}_k$ are Hermitian and at most one element of each row/column is nonzero, the adjoint representations take the form (up to some permutation of rows and columns)

$$\bar{\boldsymbol{H}}_k = 2\left(\bigoplus_{m_1=1}^{(\dim(\mathcal{L}_\lambda)-\dim(\ker(\bar{\boldsymbol{H}}_k)))/2}\bar{\boldsymbol{\sigma}}^y\right)\oplus\left(\bigoplus_{m_2=1}^{\dim(\ker(\bar{\boldsymbol{H}}_k))/2}\bar{\boldsymbol{0}}\right), \tag{B7}$$

where the factor of 2 originates from the Pauli commutation relations, and $\bar{\boldsymbol{\sigma}}^y = \begin{bmatrix}0 & -i\\ i & 0\end{bmatrix}$ and $\bar{\boldsymbol{0}} = \begin{bmatrix}0 & 0\\ 0 & 0\end{bmatrix}$ are the Pauli Y matrix and the zero matrix, respectively. Since the eigendecomposition of $\bar{\boldsymbol{\sigma}}^y$ is known, one may compute the eigendecomposition of $\bar{\boldsymbol{H}}_k$ in $\mathcal{O}(\dim(\mathcal{L}_\lambda))$ by iterating over the $\bar{\boldsymbol{\sigma}}_y$ summands, populating the corresponding eigenvectors and eigenvalues, and using any set of vectors spanning $\ker(\bar{\boldsymbol{H}}_k)$ as the remaining eigenvectors with zero eigenvalues.

Appendix C: Gradient calculations

1. Observable gradients

For VQA and QML applications, it is often beneficial to be able to efficiently calculate gradients $\boldsymbol{\nabla}_{\boldsymbol{\theta}}\langle O\rangle$ of observables $O=\sum_\alpha(\boldsymbol{w})_\alpha B_\alpha^{(\lambda)}$ for $\boldsymbol{w}\in\mathbb{R}^{\dim(\mathcal{L}_\lambda)}$, $B_\alpha^{(\lambda)}\in\mathcal{L}_\lambda$. Differentiating Eq. (14) with respect to θ_m, we find

$$(\boldsymbol{\nabla}_{\boldsymbol{\theta}}\langle O\rangle)_m=\frac{\partial\langle O\rangle}{\partial\theta_m}=\boldsymbol{w}^T\left(\frac{\partial\Phi_\lambda^{\mathrm{Ad}}(U(\boldsymbol{\theta}))}{\partial\theta_m}\right)\boldsymbol{e}_\lambda^{(\mathrm{in})}, \tag{C1}$$

where we may compute

$$\frac{\partial\Phi_\lambda^{\mathrm{Ad}}(U(\boldsymbol{\theta}))}{\partial\theta_m}=-i\left(\bar{\boldsymbol{U}}_{m+1:M}\bar{\boldsymbol{H}}_k\bar{\boldsymbol{U}}_{1:m}\right), \tag{C2}$$

where $\bar{\boldsymbol{U}}_{a:b}\equiv e^{-i\theta_b\bar{\boldsymbol{H}}_b}e^{-i\theta_{b-1}\bar{\boldsymbol{H}}_{b-1}}\dots e^{-i\theta_a\bar{\boldsymbol{H}}_a}$, m is a combined index for l and k, and $M=LK$. Naïvely evaluating Eq. (C2) requires $\mathcal{O}(M^2)$ matrix-vector multiplications (each $\mathcal{O}(\dim(\mathcal{L}_\lambda)^2)$), however by exploiting a recurrent property of the gradients, one may compute the gradient with only $\mathcal{O}(M)$ matrix-vector multiplications (i.e., with the same asymptotic complexity as simulating the circuit in the first place). This procedure, which follows the 'reverse-mode' gradient calculation of Ref. [67] in conjunction with the efficient spectral approach of Algorithm 1, is outlined in Algorithm 2.

Algorithm 2 Calculating gradients in $\mathfrak{g}$-sim.

```
Input: Circuit $\boldsymbol{\theta}$, input expectation value vector $\boldsymbol{e}^{(\mathrm{in})}$, circuit generator adjoint representations $\bar{\boldsymbol{H}}_k$ and eigendecompositions
  $\bar{\boldsymbol{Q}}_k$, $\epsilon_k^{(g)}$, coefficients $\boldsymbol{w}$ of observable $O$
  $\boldsymbol{\eta}\leftarrow\bar{\boldsymbol{U}}\boldsymbol{e}^{(\mathrm{in})}$                         ▷ Use the main loop of Algorithm 1.
  $\boldsymbol{\phi}\leftarrow\boldsymbol{w}$
  $(\boldsymbol{\nabla}_{\boldsymbol{\theta}}\langle O\rangle)_M\leftarrow-i\boldsymbol{\phi}^T\bar{\boldsymbol{H}}_M\boldsymbol{\eta}$
  for $m$ in $N,\dots,2$ do
    $\boldsymbol{\eta}\leftarrow\bar{\boldsymbol{Q}}_m\boldsymbol{\eta}$
    $\boldsymbol{\phi}\leftarrow\bar{\boldsymbol{Q}}_m\boldsymbol{\phi}$
    for $g$ in $1,\dots,\dim(\mathcal{L}_\lambda)$ do
      $\eta_g\leftarrow e^{i\theta_m\epsilon_m^{(g)}}\eta_g$
      $\phi_g\leftarrow e^{i\theta_m\epsilon_m^{(g)}}\phi_g$
    end for
    $\boldsymbol{\eta}\leftarrow\bar{\boldsymbol{Q}}_m^T\boldsymbol{\eta}$
    $\boldsymbol{\phi}\leftarrow\bar{\boldsymbol{Q}}_m^T\boldsymbol{\phi}$
    $(\boldsymbol{\nabla}_{\boldsymbol{\theta}}\langle O\rangle)_{m-1}\leftarrow-i\boldsymbol{\phi}^T\bar{\boldsymbol{H}}_m\boldsymbol{\eta}$
  end for
Output: Gradient vector $\boldsymbol{\nabla}_{\boldsymbol{\theta}}\langle O\rangle$
Asymptotic complexity: $\mathcal{O}(M\dim(\mathcal{L}_\lambda)^2)$
```

Higher order derivatives can also be computed, such as the Hessian

$$(\boldsymbol{\nabla}^2_{\boldsymbol{\theta}}\langle O\rangle)_{jk}=\frac{\partial^2\langle O\rangle}{\partial\theta_j\partial\theta_k}=-\boldsymbol{h}^T\bar{\boldsymbol{U}}_{k+1:N}\bar{\boldsymbol{H}}_k\bar{\boldsymbol{U}}_{j:k}\bar{\boldsymbol{H}}_j\bar{\boldsymbol{U}}_{1:j}\boldsymbol{e}^{(\mathrm{in})}. \tag{C3}$$

However, since the circuit generators $\bar{\boldsymbol{H}}_k$ are typically non-invertible, Algorithm 2 cannot be generalized to evaluate Eq. (C3) and higher-order derivatives. Although still polynomial in complexity, evaluating a P-th order gradient for $P\geqslant 2$ will be a factor $\mathcal{O}(M^P)$ slower than efficiently evaluating the gradient.

2. Compilation of cost function gradients

For unitary compilation (Sec. IV C), the use of gradient-based optimizers requires the efficient calculation of $\partial\mathcal{L}_{\mathfrak{g}}/\partial\theta_m$. Differentiating Eq. (40), we obtain

$$\frac{\partial\mathcal{L}_{\mathfrak{g}}}{\partial\theta_m}=\frac{1}{\dim(\mathfrak{g})}\operatorname{Im}\left[\operatorname{Tr}\left[\bar{\boldsymbol{V}}^T\bar{\boldsymbol{U}}_{m+1:M}\bar{\boldsymbol{H}}_m\bar{\boldsymbol{U}}_{1:m}\right]\right], \tag{C4}$$

where we have used properties of the ansatz structure, given in Eq. (A15), and $\bar{\boldsymbol{U}}_{n:m}^T \equiv \bar{\boldsymbol{U}}_n^T \bar{\boldsymbol{U}}_{n+1}^T \ldots \bar{\boldsymbol{U}}_m^T$. Noting that the matrix terms in Eq. (C4) have the same recurrent form as Eq. (C2), we may compute the full gradient vector in $\mathcal{O}(N \dim(\mathfrak{g})^3)$ by a straightforward modification of Algorithm 2 (with matrix-matrix multiplication instead of matrix-vector multiplication).

For efficient implementation, care should be taken with the order of operations when evaluating Eqs. (40) and (C4), in particular to avoid any dense-dense matrix multiplication. For example, in Eq. (C4), one should store the products $(\bar{\boldsymbol{U}}_{m+1:m})^T \bar{\boldsymbol{V}}$ and $\bar{\boldsymbol{U}}_{1:m}$ as dense matrices, use sparse representations of the $\bar{\boldsymbol{U}}_m$ when applying the iterative step in Algorithm 2, and use sparse representations of $\bar{\boldsymbol{H}}_m$ when evaluating each element of the gradient. We find that with this order of operations, the evaluation is substantially accelerated by the use of sparse-dense matrix multiplication algorithms, and further improved by the use of GPU acceleration (e.g. SpDM with cuSPARSE). Finally, the trace in Eq. (C4) of the product of two dense matrices ($\bar{\boldsymbol{V}}^T \bar{\boldsymbol{U}}_{m+1:M}$ and $\bar{\boldsymbol{H}}_m \bar{\boldsymbol{U}}_{1:m}$) can be evaluated in a dense $\mathcal{O}(\dim(\mathfrak{g})^2)$ operation as

$$\mathrm{Tr}\big[\bar{\boldsymbol{V}}^T \bar{\boldsymbol{U}}_{m+1:M} \bar{\boldsymbol{H}}_m \bar{\boldsymbol{U}}_{1:m}\big] = \sum_{\alpha\beta} \big[((\bar{\boldsymbol{U}}_{m+1:m})^T \bar{\boldsymbol{V}}) \circ (\bar{\boldsymbol{H}}_m \bar{\boldsymbol{U}}_{1:m})\big]_{\alpha\beta}\,, \tag{C5}$$

where $\circ$ denotes the Hadamard product. This avoids costly $\mathcal{O}(\dim(\mathfrak{g})^3)$ dense-dense matrix multiplication entirely, which would otherwise be by far the most expensive operation in this procedure.

3. Analog computing

Gradients of expectation values can also be obtained in the analog computing scenario that was laid out in Appendix A 4. While not providing full-fledged details of the implementation required, we specify the problem and point the reader toward dedicated references for implementation.

In the analog computing scenario, control of the system occurs at the Hamiltonian level through the control functions $c_h(t)$ appearing in Eq. (A37). A first step towards defining the gradients of interest requires parametrizing these control functions in terms of a finite set of parameters $\boldsymbol{\theta}$ (i.e., $c_h(t) \to c_h(\boldsymbol{\theta}, t)$). Upon such parametrization, and given an observable O, an optimization task would consist in identifying values of $\boldsymbol{\theta}$ that minimize the expectation value $\langle O \rangle$, with the gradients of interests denoted $\partial \langle O \rangle / \partial_{\boldsymbol{\theta}_m}$.

While similar in notations to the gradients of the circuit parameters in Eq. (C1), we note that depending on the parametrization of the controls adopted, each of the control parameter may affect the control values along the whole evolution, and would appear particularly difficult to evaluate. Still, progress have been made towards efficient evaluation of such gradients in the context of quantum optimal control and machine learning. In particular, methods presented in Refs. [140, 141] generically enable such calculations and require to numerical solve an augmented ODE, but avoid the necessity of storing in memory the state of the system along each numerical step of its evolution. Applications of these numerical methods have already found application in quantum optimal control problems [142, 143] but to-date have remained limited to small system sizes due to the exponentially growth of the underlying Hilbert space. Incorporating them with $\mathfrak{g}$-sim would circumvent such limitations for certain quantum dynamics.

Appendix D: Review of Wick's theorem

In this section, we review Wick's theorem from a Lie-algebraic perspective, which can be used to efficiently compute expectation values of arbitrary operators over the highest weight states of the circuit's Lie algebra [43, 44].

Consider the Cartan-Weyl decomposition of $\mathfrak{g}$, given by $\mathfrak{g} = \mathfrak{h} \oplus \mathfrak{g}_+ \oplus \mathfrak{g}_-$, where $\mathfrak{h} = \mathrm{span}_{\mathbb{R}} i\{\widetilde{H}_k\}$ denotes the Cartan subalgebra, $\mathfrak{g}_+ = \mathrm{span}_{\mathbb{R}} i\{E_{\alpha'}\}$ and $\mathfrak{g}_- = \mathrm{span}_{\mathbb{R}} i\{E_{-\alpha'}\}$, with $E_{\alpha'}$ and $E_{-\alpha'}$ being the sets of raising and lowering operators, respectively. Let us denote as $|\mathrm{hw}\rangle$ the highest weight state of the algebra. In what follows, we will study the task of simulating an expectation value $\langle O(\boldsymbol{\theta})\rangle \equiv \mathrm{Tr}\big[O U(\boldsymbol{\theta}) |\mathrm{hw}\rangle\langle\mathrm{hw}| U^\dagger(\boldsymbol{\theta})\big]$. Note that more generally, we could also consider the case when the initial state is a generalized coherent state [70–73], i.e., a state of the form $|\psi\rangle = V|\mathrm{hw}\rangle$ with some (known) $V \in e^{\mathfrak{g}}$ as the action of V can be absorbed into $U(\boldsymbol{\theta})$.

To begin, take $O \in i\mathfrak{g}$. Then, as per Eq. (10) we know that $U^\dagger(\boldsymbol{\theta}) O U(\boldsymbol{\theta})$ will always be expressed as

$$U^\dagger(\boldsymbol{\theta}) O U(\boldsymbol{\theta}) = \sum_k c_k \widetilde{H}_k + \sum_{\alpha'} c_{\alpha'} E_{\alpha'} + c^*_{-\alpha'} E_{\alpha'}\,, \tag{D1}$$

where the coefficients c_k, $c_{\alpha'}$ and $c^*_{\alpha'}$ can be obtained from the adjoint representation of the unitary. Since

$$E_{\alpha'}|\mathrm{hw}\rangle = \langle\mathrm{hw}| E_{-\alpha'} = 0, \quad \widetilde{H}_k |\mathrm{hw}\rangle = \widetilde{h}_k |\mathrm{hw}\rangle\,, \tag{D2}$$

where $\widetilde{h}_k$ are the weights of $|\mathrm{hw}\rangle$, we can see that

$$\langle O(\boldsymbol{\theta})\rangle = \sum_k c_k \widetilde{h}_k \,. \tag{D3}$$

Note that up to this point, we have used the adjoint representation (just like in $\mathfrak{g}$-sim), but we have leveraged the fact that the expectation values of a highest-weight on the algebra elements can be readily computed via the Cartan-Weyl decomposition. This is the key idea behind Wick's theorem.

Next, consider more general observables given by products of (polynomially-many) elements of the algebra such as $O = O_1 O_2$, with $O_1, O_2 \in i\mathfrak{g}$. These can belong to higher dimensional irreps, making a simulation via $\mathfrak{g}$-sim more expensive according to Eq. (1). However, we can again use the adjoint representation to evolve each element in the algebra as $U^\dagger(\boldsymbol{\theta}) O U(\boldsymbol{\theta}) = U^\dagger(\boldsymbol{\theta}) O_1 U(\boldsymbol{\theta}) U^\dagger(\boldsymbol{\theta}) O_2 U(\boldsymbol{\theta})$, which will lead to a summation of elements of the Cartan subalgebra, times raising and lowering root operators. Given that these act on the highest-weight, only a few of those terms will be non-zero. In the former case, only the terms $\langle \widetilde{H}_k \widetilde{H}_{k'}\rangle = \widetilde{h}_k \widetilde{h}_{k'}$, and $\langle E_{\alpha'} E_{-\alpha'}\rangle$ are non-zero. The latter can be expressed in terms of components of the root vector (see [43]).

More generally, the overall goal of Wick's theorem is to use the (known) commutation relations of the elements of the algebra to get all the lowering operator to the "left" and all the raising operators to the "right", as these will vanish when acting on the highest-weight state. While this commutation procedure could be computationally demanding in principle, in practice the properties of $\mathfrak{g}$ can be used to greatly simplify the required calculations. For instance, when considering free-fermionic evolutions (see the main text), the highest weight, and generalized coherent states, correspond to Gaussian states and the evaluation of expectation values over products of elements in the algebra can be mapped to the task of evaluating a Pfaffian [144]. Hence, the simulation of expectation values for free-fermionic evolutions becomes efficient when the initial state is a Gaussian, or a linear combination of few Gaussian states.

When applying Wick's theorem to more general initial states, one would need to expand them into a linear combination of generalized coherent states, and apply Wick's theorem for each one, as well as for the cross terms. In such case, the computational complexity scales with the number of generalized coherent states, and when such number is exponential, the overall cost becomes exponential, even for observables in $i\mathfrak{g}$.

Appendix E: Ansatz circuits and benchmarking

Details of all ansatz circuits used in this work are given in Table I.

1. Benchmark calculations

For the compute time benchmarks in Fig. 3, we repeatedly simulate evolution of a state initialized to $|\mathbf{0}\rangle$ and transformed by a single layer ($L = 1$) of the TFXY 2-local ansatz (see Table I) using both our Python implementation of $\mathfrak{g}$-sim and TensorFlow Quantum on a single core of an Intel Xeon Platinum 8268. We stress that unlike tensor network methods, the time complexity of applying a gate is independent of circuit depth, so we report our timings per gate. We collect timings using the `timeit` Python package in batches of repeated simulations (to minimize the influence of measurement distortion due to environmental factors) and average over batches. We bootstrap 95% confidence intervals on these averages, but they are too small to be visualized in Fig. 3.

Memory requirements for state vectors are computed assuming double-precision arithmetic on a 2^n-dimensional Hilbert space $\mathcal{H}$. Memory requirements for $\mathfrak{g}$-sim are computed by profiling the actual memory footprint of the observable vector $\boldsymbol{e}$ and algebra data (adjoint representations of generators $\bar{\boldsymbol{H}}_k$ and their eigendecompositions).

Appendix F: Subalgebras of $\mathfrak{g}_0$

Throughout this work, we use the algebra $\mathfrak{g}_0$, defined in Eq. (22), which has dimension $\dim(\mathfrak{g}_0) = n(2n-1)$ and encompasses many models of interest:

- The Hamiltonian algebra for the XY model with open boundary conditions and free coefficients forms a proper subalgebra $\mathfrak{g}_{\mathrm{XY}} \subset \mathfrak{g}_0$ with $\dim(\mathfrak{g}_{\mathrm{XY}}) = n(n-1)$ [110], given by

$$\mathfrak{g}_{\mathrm{XY}} = \mathrm{span}_{\mathbb{R}} \left\langle \left(\{i\sigma_j^x \sigma_{j+1}^x\}_{j=1}^{n-1}\right) \cup \left(\{i\sigma_j^y \sigma_{j+1}^y\}_{j=1}^{n-1}\right)\right\rangle_{\mathrm{Lie}} . \tag{F1}$$

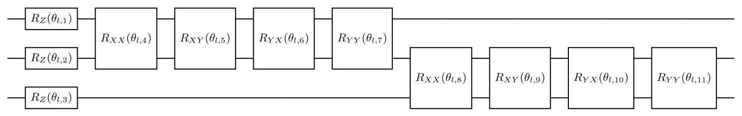
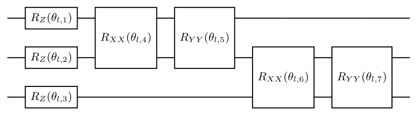
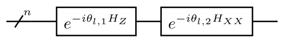
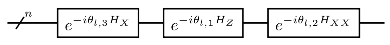
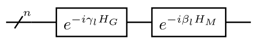
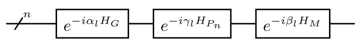
	Applications	$\mathfrak{g}$-sim efficient	Ansatz layer
TFXY 2-local	Benchmarking (B 1) TFXY VQE (IV A) Random compilation (IV C 2) Phase classifier (IV D)	Yes	
TFXY Trotter	Dynamics compilation (IV C 4)	Yes	
TFXY HVA	LTFIM pre-training (IV B 2)	Yes	
LTFIM HVA	LTFIM pre-training (IV B 2)	No	
QAOA standard	QAOA pre-training (IV B 3)	For $H_G \in \{P_n, C_n\}$	
QAOA pre-trained	QAOA pre-training (IV B 3)	For $H_G \in \{P_n, C_n\}$	

TABLE I. Layers of all variational ansätze used in this work, with reference to their applications, whether or not they are classically efficient in $\mathfrak{g}$-sim (for simulating dynamics of observables supported by $\mathfrak{g}$), and circuit diagrams for single layers of each ansatz. In diagrams where individual qubits are depicted, the example layer is given for $n = 3$ for compactness.

- The Hamiltonian algebra for the transverse field Ising model (TFIM) with open boundary conditions and non-free coupling coefficients forms a proper subalgebra $\mathfrak{g}_{\text{TFIM}} \subset \mathfrak{g}_0$ with $\dim(\mathfrak{g}_{\text{TFIM}}) = n^2$ [78], given by

$$\mathfrak{g}_{\text{TFIM}} = \text{span}_{\mathbb{R}} \left\langle \left(i \sum_{j=1}^{n-1} \sigma_j^x \sigma_{j+1}^x \right) \cup \left(i \sum_{j=1}^{n} \sigma_j^z \right) \right\rangle_{\text{Lie}} . \tag{F2}$$

- For a path graph P_n on n vertices, the circuit of the quantum approximate optimization algorithm [99] has an associated algebra

$$\mathfrak{g}_{\text{QAOA},P_n} = \text{span}_{\mathbb{R}} \left\langle \left(i \sum_{j=1}^{n-1} \sigma_j^z \sigma_{j+1}^z \right) \cup \left(i \sum_{j=1}^{n} \sigma_j^x \right) \right\rangle_{\text{Lie}} , \tag{F3}$$

  which is isomorphic to the one of the TFIM, $\mathfrak{g}_{\text{QAOA},P_n} \cong \mathfrak{g}_{\text{TFIM}}$. The two are related by a Hadamard transform.

Appendix G: Further details on circuit compilation

1. Compiling small-T global rotations

In Sec. IV C 2, we investigate the trainability of $\mathcal{L}_{\mathfrak{g}}$ for general random $V \in \mathcal{G}$. This includes both global and local dynamics (the respective locality is controlled by the choice of Hamiltonian generators G_α that are included in Eq. (41)). Our numerics suggest that compilation of local dynamics is scalable in system size n and rotation angle (dynamics duration) T, with $\mathcal{L}_{\mathfrak{g}}$ vanishing exponentially in the number of iterations at all tested values for local dynamics. Similarly, for global dynamics, $\mathcal{L}_{\mathfrak{g}}$ vanishes exponentially in the number of iterations when T is sufficiently large (beyond $T \approx 1$ for a normalized global Hamiltonian). However, for smaller values of T, a different convergence behavior is seen when compiling global dynamics. After a brief initial period where $\mathcal{L}_{\mathfrak{g}}$ vanishes exponentially in the

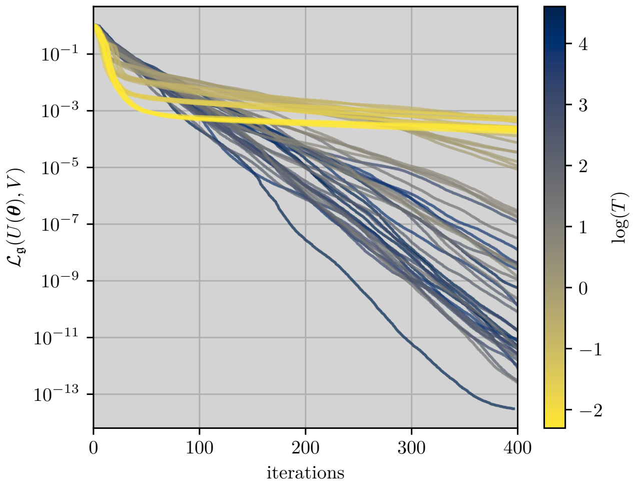


FIG. 12. **Compiling short-time dynamics for global Hamiltonians.** For a system of $n = 20$ qubits, we train an overparametrized (27-layer) ansatz to minimize $\mathcal{L}_{\mathfrak{g}}$, with a target unitary generated by a normalized global Hamiltonian. At larger values of T, the expected exponential convergence is observed. At $T \approx 1$, a transition is observed to a regime with poorer trainability.

number of iterations, a transition to very slow convergence is observed. This is demonstrated in Fig. 12. The reason for this phenomenon is not currently understood.

2. Center of $\mathcal{G}_{\mathrm{TFXY}}$

In Section IV C 3, we observed that when $Z(\mathcal{G})$ is nontrivial (i.e., including non-identity terms), elements of $Z(\mathcal{G})$ can be compiled by the ansatz but cannot be distinguished by the adjoint-representation loss function $\mathcal{L}_{\mathfrak{g}}$. Here we show that there is exactly one non-trivial element of $Z(\mathcal{G}_{\mathrm{TFXY}})$ up to a global phase.

We note that for a dynamical Lie algebra $\mathfrak{g}$ and its dynamical Lie group $\mathcal{G} = e^{\mathfrak{g}}$, we have $Z(\mathcal{G}) = \mathfrak{g}' \cap \mathcal{G}$, where $\mathfrak{g}' \equiv \{A \in \mathbb{C}^{d\times d} \mid \forall B \in \mathfrak{g}, AB = BA\}$ is the commutant of $\mathfrak{g}$. Hence, given characterization of the commutant of the Lie algebra of interest, it is relatively easy to determine the center of its Lie group. For instance, we have $\mathfrak{g}'_{\mathrm{TFXY}} = \mathrm{span}\{I, (\sigma^z)^{\otimes n}\}$ [78], and one can see that the non-identity element of the commutant $(\sigma^z)^{\otimes n}$ can be compiled up to a global phase, since $\sigma_z = ie^{-\frac{i\pi}{2}\sigma_z} = -ie^{-\frac{3i\pi}{4}\sigma_z}$. That is, $(\sigma^z)^{\otimes n}$ can be compiled with σ_z rotations on each qubit, with angles $\pi/2$ or $\pi/4$, and thus belong to $\mathcal{G}_{\mathrm{TFXY}}$. This leaves us with exactly one (non-trivial) element of the group that cannot be detected by $\mathcal{L}_{\mathfrak{g}}$.

Appendix H: Implementing the binary quantum-phase classifier on a quantum device

In Section IV D, we described the training of a quantum-phase classifier, implemented via a variational ansatz circuit $U(\boldsymbol{\theta})$ defined in Eq. (42). After training, a set of optimal parameters $\boldsymbol{\theta}^*$ is acquired, allowing classification of unseen states using the unitary $U_* \equiv U(\boldsymbol{\theta}^*)$. Straightforward implementation of the classifier on quantum hardware consists in applying the unitary U_* to an unknown state, measuring the operator $O = \sigma_1^z$, and assigning a label according to Eq. (46). However we could also replicate the model by means of measurements and post-processing only. Indeed, while Result 1 was framed as an evolution of expectation values, it can equivalently be seen as an (Heisenberg) evolution of the observable.

Given the adjoint representation $\bar{\boldsymbol{U}}_* = \Phi_\lambda^{\mathrm{Ad}}(U_*)$ of the trained circuit (computed in $\mathfrak{g}$-sim), and a description of the observable $O \in i\mathfrak{g}$ as the vector $\boldsymbol{w}$ such that $O = \sum(\boldsymbol{w})_\alpha G_\alpha$, we have

$$O' := U_*^\dagger O U_* = \sum_{\alpha=1}^{\dim(\mathfrak{g})} (\boldsymbol{w}')_\alpha G_\alpha, \text{ with } \boldsymbol{w}' = \boldsymbol{w}^{\mathrm{T}} \bar{\boldsymbol{U}}_*. \tag{H1}$$

Hence, one recovers the classifier outputs as a weighted sum of the expectation values of the observables G_α.

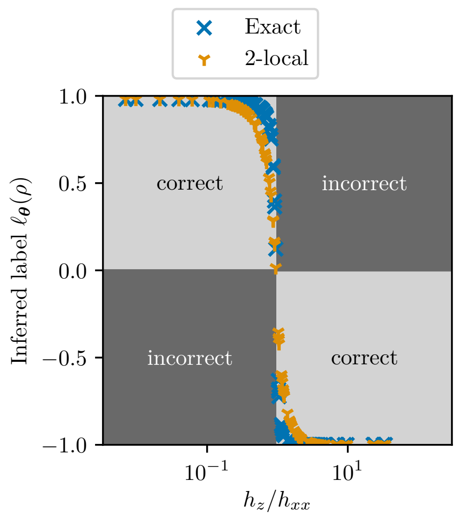


FIG. 13. **Comparison of exact label inferences to their 2-local approximation.** We compare the exact inferred labels of the TFIM phase classifier on an unseen training dataset to a 2-local approximation. The 2-local approximation still achieves 100% classification accuracy, although $\ell_{\boldsymbol{\theta}}(\rho)$ changes less rapidly near the phase transition.

Depending on the characteristics of the quantum device at hand (i.e., measurement versus gate fidelity and cycle times), one approach may be preferred to another. For the measurement-based one, one may further reduce the amount of measurements required by pruning the model (i.e., approximating Eq. (H1) with a subset of the terms involved). For instance, this could be achieved by discarding terms corresponding to small weights $|(\boldsymbol{w}')_\alpha| < \delta$ for some threshold δ, or by discarding greater-than-k-local terms. In particular, the later could be combined with classical shadows [145] by means of random measurements (rather than direct measurements of expectation values of all the G_α) ensuring small approximation error due to the bounded locality.

To study further the effect of such locality-pruning, we compare the accuracy of a trained classifier to its 2-local approximation, for an undisguised ($T = 0$) TFIM of $n = 50$ qubits. As can be seen in Figure 13, despite slight differences in the output of the models, the approximated classifier retains full accuracy (with 100% of the testing data correctly classified) that can be explained by the high locality of the TFIM order parameters.