Here comes the SU(N): multivariate quantum gates and gradients

Roeland Wiersema[1,2], Dylan Lewis[3], David Wierichs[4], Juan Carrasquilla[1,2], and Nathan Killoran[4]

[1]Vector Institute, MaRS Centre, Toronto, Ontario, M5G 1M1, Canada

[2]Department of Physics and Astronomy, University of Waterloo, Ontario, N2L 3G1, Canada

[3]Department of Physics and Astronomy, University College London, London WC1E 6BT, United Kingdom

[4]Xanadu, Toronto, ON, M5G 2C8, Canada

Variational quantum algorithms use non-convex optimization methods to find the optimal parameters for a parametrized quantum circuit in order to solve a computational problem. The choice of the circuit ansatz, which consists of parameterized gates, is crucial to the success of these algorithms. Here, we propose a gate which fully parameterizes the special unitary group $\mathrm{SU}(N)$. This gate is generated by a sum of non-commuting operators, and we provide a method for calculating its gradient on quantum hardware. In addition, we provide a theorem for the computational complexity of calculating these gradients by using results from Lie algebra theory. In doing so, we further generalize previous parameter-shift methods. We show that the proposed gate and its optimization satisfy the quantum speed limit, resulting in geodesics on the unitary group. Finally, we give numerical evidence to support the feasibility of our approach and show the advantage of our gate over a standard gate decomposition scheme. In doing so, we show that not only the expressibility of an ansatz matters, but also how it's explicitly parameterized.

1 Introduction

Variational quantum computing is a paradigm of quantum computing that uses optimization algorithms to find the optimal parameters for a parameterized quantum circuit [1, 2]. Crucial for the success of such algorithms is the choice of circuit ansatz, which usually consists of multiple parameterized one and two-qubit gates. Typi-cally, these gates are parameterized unitary matrices generated by single Pauli-string operators that can locally rotate a state around some axis: $U(t) = \exp\{itG\}$, where t is a gate parameter and G a Pauli string. For a specific family of cost functions, there exist a variety of methods that allow one to obtain the gradient with respect to t [3, 4, 5, 6, 7, 8, 9] on quantum hardware. With these gradients, the cost function can be minimized via any gradient-descent-based algorithm.

Instead of considering a gate generated by a single Pauli string, one can construct more general parameterized gates that can perform an arbitrary rotation in $\mathrm{SU}(N)$, the special unitary group. These general $\mathrm{SU}(N)$ rotations are used in a variety of quantum algorithms [10, 11, 12, 13]. In practice, rotations in $\mathrm{SU}(N)$ can be implemented by composing several simple parameterized gates together into a more complicated one. For example, for single and two-qubit gates (where $N = 2, 4$, respectively), there exist several general decomposition schemes of such gates into products of single-qubit gates and CNOTs [14, 15, 16, 17, 18, 19]. In practice, this compilation comes with hardware-specific challenges, since quantum hardware usually has a set of native gates into which all others have to be decomposed [20, 21].

Choosing the right parameterization for a function is important because it can significantly affect the properties of its gradients. Reparameterizing functions to obtain more useful gradients is a well-known method in statistics and machine learning. For example, in restricted maximum likelihood methods one can ensure numerical stability of quasi-Newton methods by decomposing covariance matrices into Cholesky factors [22]. In addition, methods like auxiliary lin-

ear transformations [23], batch normalization [24] and weight normalization [25] are used to improve the gradients in neural networks. In variational inference, the reparameterization trick [26] is at the core of variational autoencoder approaches and allows for gradients for stochastic back-propagation [27, 28]. In light of this, it may be worthwhile to investigate alternative parameterizations of quantum gates for variational quantum algorithms.

In this work, we propose a family of parameterized unitaries called $\mathbb{SU}(N)$ gates and provide a method to evaluate their gradients on quantum hardware. In doing so, we generalize the prior literature one step further, since many past schemes can be understood as special cases of our proposal [3, 4, 5, 6, 7, 8, 9]. We provide numerical results to support the validity of our approach and give several examples to illustrate the capabilities of the $\mathbb{SU}(N)$ gate. We show that this gate satisfies the quantum speed limit and that it is easier to optimize compared to $\mathrm{SU}(N)$ parameterizations that consist of products of gates. We argue that this is the case because the product of unitaries creates a "bias" in the Lie algebra that deforms the cost landscape. In addition, we highlight the connections between our formalism and the properties of semisimple Lie algebras and establish a bound on the computational complexity of the gradient estimation using tools from representation theory.

2 $\mathbb{SU}(N)$ gates

A quantum gate is a unitary operation U that acts on a quantum state ρ in a complex Hilbert space. If we ignore a global phase, then a gate U acting on N_{qubits} qubits is an element of the special unitary group $\mathrm{SU}(N)$ (see App. A), where $N = 2^{N_{\mathrm{qubits}}}$. Note that all of the following works for any $N > 1$, but here we restrict ourselves to the qubit case. We are interested in constructing a quantum gate that parameterizes all of $\mathrm{SU}(N)$. To achieve this, we make use of the theory of Lie algebras. We will not be concerned with the formal treatment of this topic, which can be found in many excellent textbooks [29, 30, 31].

To construct our gate, we realize that $\mathrm{SU}(N)$ is a (semisimple) Lie group and so there exists a unique connection between its elements and the Lie algebra $\mathfrak{su}(N)$ via the so-called Lie corre-

spondence, or Lie's third theorem [32, 31]. In particular, each $g \in \mathrm{SU}(N)$ can be identified with an $A \in \mathfrak{su}(N)$ via the exponential map $g = \exp\{A\}$. For our purposes, we can understand the Lie algebra $\mathfrak{su}(N)$ as a vector space of dimension $N^2 - 1$ that is closed under the commutator, $[A, B] = AB - BA \in \mathfrak{su}(N)$ for $A, B \in \mathfrak{su}(N)$. For $\mathfrak{su}(N)$, we choose as a basis the tensor products of Pauli matrices multiplied by the imaginary unit i:

$$\mathcal{P}^{(N_{\mathrm{qubits}})} = \left\{ i(\sigma_1 \otimes \ldots \otimes \sigma_{N_{\mathrm{qubits}}}) \right\} \setminus \{I_{N_{\mathrm{qubits}}}\}, \quad (1)$$

where $\sigma_i \in \{I, X, Y, Z\}$ and $I_{N_{\mathrm{qubits}}} = iI^{\otimes N_{\mathrm{qubits}}}$.

We choose the following parameterization of $\mathrm{SU}(N)$:

$$U(\boldsymbol{\theta}) = \exp\{A(\boldsymbol{\theta})\}, \quad A(\boldsymbol{\theta}) = \sum_m \theta_m G_m, \quad (2)$$

where $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_{N^2-1}) \in \mathbb{R}^{N^2-1}$ and $\{G_m\} \in \mathcal{P}^{(N_{\mathrm{qubits}})}$. To distinguish between the group and the gate, we call the parameterization in Eq. (2) a $\mathbb{SU}(N)$ gate. The coordinates $\boldsymbol{\theta}$ are called the canonical coordinates, which uniquely parameterize U through the Lie algebra $\mathfrak{su}(N)$. Since we typically cannot implement the above gate in hardware, we will have to be decompose it via a standard unitary decomposition algorithm [14, 15, 16, 17, 18, 19]. We emphasize here that even though the gate will be decomposed, it is parameterized as an exponential map. Hence we can understand Eq. (2) as a change of coordinates from the $\mathrm{SU}(N)$ gate decomposition.

If we do not want to parameterize all of $\mathrm{SU}(N)$, we can instead parameterize a more restricted Hamiltonian by setting some of the parameters θ_m to zero. This makes Eq. (2) a natural parameterization of several Hamiltonians available on modern quantum hardware platforms. These Hamiltonians often have multiple independently tunable fields which can be active at the same time and do not necessarily commute. One typically has local control on each qubit and access to an interacting Hamiltonian between pairs of qubits, depending on the topology of the quantum device [33]. The interacting pair can for example be a ZZ interaction for Josephson flux qubits [34], a Heisenberg interaction for nuclear spins in doped silicon [35] or an XY interaction in quantum dots interacting with a cavity [36].

To use this gate in a gradient-based variational quantum algorithm, we have to be able to obtain partial derivatives of $U(\boldsymbol{\theta})$ with respect to each parameter θ_l. Although there exist a variety of works that provide analytical expressions for gradients through quantum circuits via the parameter-shift rule [3, 4, 5, 6, 7, 8, 9, 37], these works almost uniformly assume that the gate is of the form $U(\theta) = \exp\{i\theta P\}$, where P is a Hermitian operator. As far as we are aware, the only methods to obtain gradients of Eq. (2) with respect to $\boldsymbol{\theta}$ are the stochastic and Nyquist parameter-shift rules of [6] and [10], respectively. The first approach relies on an integral identity for bounded operators that is estimated via Monte Carlo [38], whereas the latter is based on a theorem in Fourier analysis [39].

3   Obtaining the gradient

Here, we provide a new approach to obtain the gradient of Eq. (2) that makes use of differentiable programming, which is efficient for gates acting on a small number of qubits. To start, we note that the partial derivative with respect to a parameter θ_l is given by

$$
\begin{aligned}
\frac{\partial}{\partial \theta_l} U(\boldsymbol{\theta}) &= \frac{\partial}{\partial \theta_l} \exp\{A(\boldsymbol{\theta})\} \\
&= U(\boldsymbol{\theta}) \sum_{p=0}^{\infty} \frac{(-1)^p}{(p+1)!} (\mathrm{ad}_{A(\boldsymbol{\theta})})^p \frac{\partial}{\partial \theta_l} A(\boldsymbol{\theta}).
\end{aligned}
$$
(3)

Here, ad_X denotes the adjoint action of the Lie algebra given by the commutator $\mathrm{ad}_X(Y) = [X, Y]$ [31]. Furthermore, we write $(\mathrm{ad}_X)^p(Y) = [X, [X, \ldots [X, Y]]]$, hence $(\mathrm{ad}_X)^p$ denotes a nested commutator of p terms. For more details, see App. B. Note that the term on the right of $U(\boldsymbol{\theta})$ in Eq. (3) is an element of the Lie algebra, since $\partial/\partial \theta_l A(\boldsymbol{\theta}) = G_l \in \mathfrak{su}(N)$ and so the commutator keeps the entire sum in the algebra. For notational clarity we define

$$
\Omega_l(\boldsymbol{\theta}) = \sum_{p=0}^{\infty} \frac{(-1)^p}{(p+1)!} (\mathrm{ad}_{A(\boldsymbol{\theta})})^p \frac{\partial}{\partial \theta_l} A(\boldsymbol{\theta}), \quad (4)
$$

where $\Omega_l(\boldsymbol{\theta}) \in \mathfrak{su}(N)$ is a skew-Hermitian operator that generates a unitary, which we call the *effective generator*. Given that Eq. (4) is an infinite series of nested commutators it is not clear

how $\Omega_l(\boldsymbol{\theta}) \in \mathfrak{su}(N)$ can be calculated in practice without truncating the sum.

We can think of U as a function $U : \mathbb{R}^{N^2-1} \to \mathrm{SU}(N)$ that we evaluate at the point $\boldsymbol{\theta}$. Since $\mathrm{SU}(N)$ is a differentiable manifold, we can define a set of local coordinates on the group and represent $U(\boldsymbol{x})$ as a matrix described by $N^2 - 1$ real numbers. Hence, we can think of our gate as a coordinate transformation between the parameters $\boldsymbol{x}$ and the entries of the matrix representing the unitary. Since $U(\boldsymbol{x})$ depends smoothly on x_l via the matrix exponential, this coordinate transformation comes with a corresponding Jacobian (or more accurately, pushforward) $dU(\boldsymbol{x}) : T_{\boldsymbol{x}}\mathbb{R}^{N^2-1} \to T_{U(\boldsymbol{x})}\mathrm{SU}(N)$ that maps vectors tangential to $\mathbb{R}^{N^2-1}$ to vectors tangential to $\mathrm{SU}(N)$. We can obtain this Jacobian by differentiating the elements $U_{nm}(\boldsymbol{x})$ with respect to x_l:

$$
\frac{\partial}{\partial x_l} U_{nm}(\boldsymbol{x}) = \partial_{x_l} \mathfrak{Re}[U_{nm}(\boldsymbol{x})] + i\partial_{x_l} \mathfrak{Im}[U_{nm}(\boldsymbol{x})].
$$
(5)

To obtain the above matrix function numerically, we rely on the fact that the matrix exponential and its derivative are implemented in differentiable programming frameworks such as JAX [40], PyTorch [41] and Tensorflow [42] through automatic differentiation. Here we make use of the JAX implementation, which provides the matrix exponential through a differentiable Padé approximation [43, 44].

Continuing, we note that evaluating $\partial U(\boldsymbol{x})/\partial x_l$ at a point $\boldsymbol{\theta}$ produces an element of the tangent space $T_{U(\boldsymbol{\theta})}\mathrm{SU}(N)$. We can move from the tangent space to the Lie algebra by left multiplying the elementwise derivative of Eq. (5) in Eq. (3) with $U^\dagger(\boldsymbol{\theta})$ (see App. A),

$$
U^\dagger(\boldsymbol{\theta}) \left(\left.\frac{\partial}{\partial x_l} U(\boldsymbol{x})\right|_{\boldsymbol{\theta}} \right) = U^\dagger(\boldsymbol{\theta}) U(\boldsymbol{\theta}) \Omega_l(\boldsymbol{\theta}) = \Omega_l(\boldsymbol{\theta}),
$$
(6)

which allows us to obtain $\Omega_l(\boldsymbol{\theta})$ exactly, up to machine precision. We emphasize that these steps can be performed on a classical computer, with a cost that is only dependent on the number of qubits the gate acts on, not the number of qubits in the circuit.

We now make the following observation: $\Omega_l(\boldsymbol{\theta})$ corresponds to a tangent vector on $\mathrm{SU}(N)$ and generates the one-parameter subgroup $V(t) =$

$\exp\{t\Omega_l(\boldsymbol{\theta})\}$ such that

$$\Omega_l(\boldsymbol{\theta}) = \frac{d}{dt}\exp\{t\Omega_l(\boldsymbol{\theta})\}\big|_{t=0} \qquad (7)$$

and

$$\frac{\partial}{\partial\theta_l}U(\boldsymbol{\theta}) = U(\boldsymbol{\theta})\frac{d}{dt}\exp\{t\Omega_l(\boldsymbol{\theta})\}\big|_{t=0}. \qquad (8)$$
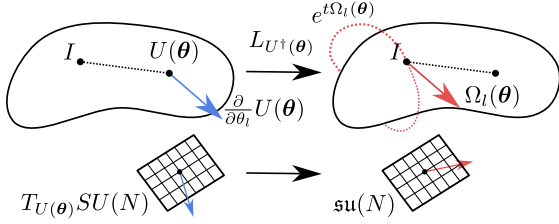
We sketch this procedure schematically in Fig. 1.

Figure 1: Schematic depiction of our approach. We move to the Lie algebra from the tangent space by left multiplication with $U^\dagger(\boldsymbol{\theta})$ and obtain $\Omega_l(\boldsymbol{\theta})$. The orbit generated by $\Omega_l(\boldsymbol{\theta})$ corresponds to the gate we have to insert in the circuit to compute the gradient.

We now consider a typical variational setting, where we are interested in minimizing the following cost function:

$$C(\boldsymbol{\theta}) = \mathrm{Tr}\Big\{U(\boldsymbol{\theta})\rho U^\dagger(\boldsymbol{\theta})H\Big\}, \qquad (9)$$

where H is some Hermitian operator and ρ the initial state of the system. For simplicity we consider a circuit consisting of a single $\mathbb{SU}(N)$ gate. Differentiating the cost function with respect to θ_l gives

$$\frac{\partial}{\partial\theta_l}C(\boldsymbol{\theta}) = \mathrm{Tr}\Big\{\Big(\frac{\partial}{\partial\theta_l}U(\boldsymbol{\theta})\Big)\rho U^\dagger(\boldsymbol{\theta})H\Big\} + \mathrm{h.c.} \qquad (10)$$

Then, plugging in Eq. (8) we find,

$$\begin{aligned}\frac{\partial}{\partial\theta_l}&C(\boldsymbol{\theta}) = \\ &\frac{d}{dt}\mathrm{Tr}\Big\{\Big(U(\boldsymbol{\theta})e^{t\Omega_l(\boldsymbol{\theta})}\rho e^{-t\Omega_l(\boldsymbol{\theta})}U^\dagger(\boldsymbol{\theta})\Big)H\Big\}\Big|_{t=0},\end{aligned} \qquad (11)$$

where we used the skew-Hermitian property of the tangent vector $\Omega_l^\dagger(\boldsymbol{\theta}) = -\Omega_l(\boldsymbol{\theta})$. Note that Eq. (11) corresponds to a new circuit with the gate $\exp\{t\Omega_l(\boldsymbol{\theta})\}$ inserted before $U(\boldsymbol{\theta})$ (see Fig. 2).
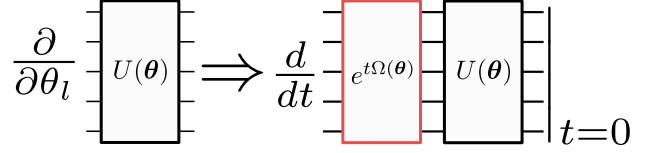

Figure 2: The partial derivative with respect to the gate parameter θ_l can be obtained by adding a gate to the circuit that is generated by $\Omega_l(\boldsymbol{\theta})$. Calculating the derivative with respect to t and evaluating at $t=0$ then provides one with the correct gradient.

The gradient of this new circuit can be computed on quantum hardware with a generalized parameter-shift rule (GPSR) [8, 7, 9]. In Algorithm 1, we outline the entire algorithm for our gradient estimation and we denote the GPSR subroutine with gpsr. An alternative to the generalized shift rule is to decompose the effective generators and apply the original two-term parameter-shift rule to the constituents (see App. 3 for details). In [45], the authors proposed the so-called stochastic parameter-shift rule for multivariate gates, which is based on the Monte Carlo approximation of an operator identity.

In Fig. 3 we consider a toy example using a random Hamiltonian on a single qubit and compare the exact derivative of an $\mathbb{SU}(2)$ gate with our generalized parameter-shift method (Algorithm 1), the stochastic parameter-shift rule and the central finite difference derivative with shifts $\pm\frac{\delta}{2}$. In particular, we consider the gate $U(\boldsymbol{\theta}) = \exp(iaX + ibY)$ with $\boldsymbol{\theta} = (a, b)$ and compute the partial derivative with respect to a over the range $a \in [0, \pi]$ for three fixed values of b on a state vector simulator (without shot noise). For the finite difference recipe we use $\delta = 0.75$, which we found to be a reasonable choice for a shot budget of 100 shots per cost function evaluation (see App. 2). We observe that the generalized SU(N) derivative reproduces the exact value while the finite difference derivative is slightly biased. This is to be expected because the latter is an approximate method. While decreasing the shift size δ reduces the deterministic approximation error, it leads to larger overall estimation errors in shot-based computations like on quantum computers (see App. 2 and e.g., [46]). Finally, the stochastic parameter-shift rule yields an unbiased estimator for the exact derivative but has a finite variance, which we estimated using 100 samples (see App. 1). We stress that this variance is a property of the differentiation method itself and not due to

Accepted in ⟨ ⟩uantum 2024-02-28, click title to verify. Published under CC-BY 4.0.

4

sampling on the quantum computer. All methods require two unique circuits per derivative but the stochastic shift rule needs additional circuits in order to suppress the variance. We provide the code for all our numerical experiments at [47].
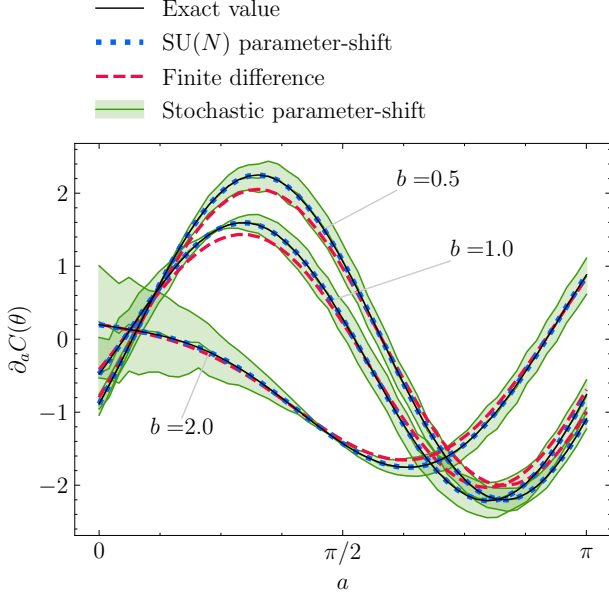

Figure 3: Gradients of $C(\boldsymbol{\theta})$ for a single $\mathbb{SU}(2)$ gate and a random single-qubit Hamiltonian, in the limit of infinitely many shots on quantum hardware. We take $A(\boldsymbol{\theta}) = iaX + ibY$ where $\boldsymbol{\theta} = (a, b)$ and consider the fixed values $b = 0.5, 1.0, 2.0$ together with $a \in [0, \pi]$. Our generalized shift rule (dotted) reproduces the exact value (solid), whereas the central finite difference (dashed) is biased and the stochastic shift rule (solid, shaded) comes with a finite statistical error even without shot noise from the quantum measurements. Since we look at a single-qubit operation, $\Omega_a(\boldsymbol{\theta})$ has a single spectral gap, so we require two shifted circuits to calculate the gradient entry (see App. D for details). The finite difference and the stochastic shift rule require two circuits as well, but additional executions are need for the latter to reduce the shown single-sample error.

In addition, we compare the three methods in the presence of shot noise in Fig. 4. We show the means and *single-shot* errors estimated with 1000 shots, which we split over 100 samples for the stochastic shift rule. We observe that the generalized SU(N) shift rule systematically performs best. It is not only unbiased but also has the smallest variance. Note that for smaller parameters b, the SU(N) shift rule and the stochastic shift rule show very similar variances. This is because $U(\boldsymbol{\theta})$ approaches the gate $R_X(a) = \exp(iaX)$, which can be differentiated with the original parameter-shift rule, and both rules in-
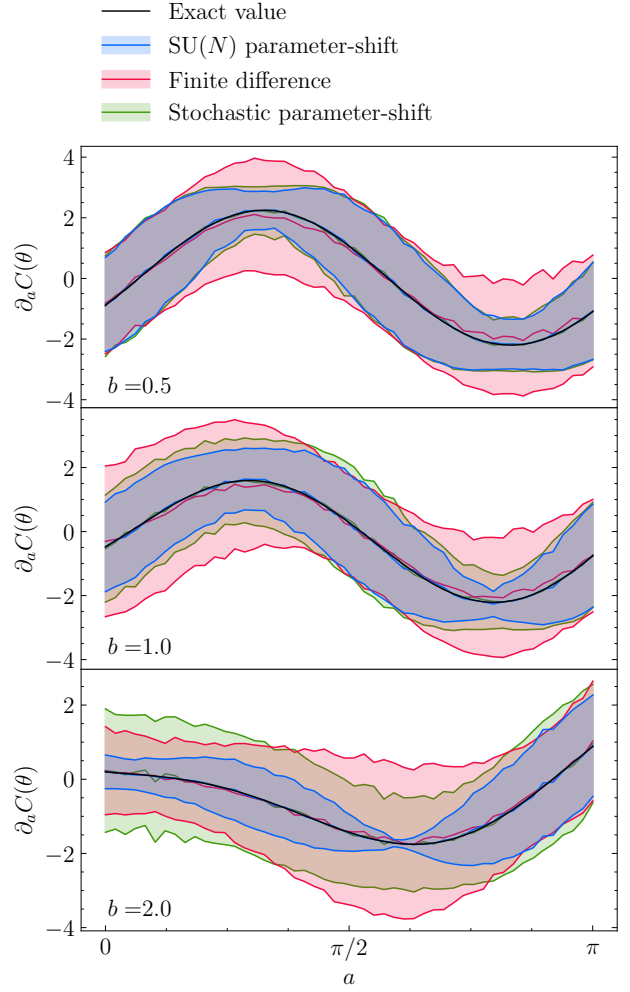
deed reduce to the two-term shift rule for R_X.

Figure 4: Gradients of $C(\boldsymbol{\theta})$ as in Fig. 3 but for finitely many shots on quantum hardware. We show the single-shot error for each method, estimated with 1000 shots, which varies with the gate parameters as noted e.g., in [9]. Our generalized SU(N) shift rule systematically outperforms the other methods. For small b, the SU(N) and the stochastic shift rule approach the single-parameter shift rule and hence behave similarly. The finite difference shift $\delta = 0.75$ is chosen such that the bias and variance are traded off reasonably for 100 shots (see App. 1 and e.g., [46]). For other shot numbers, δ needs to be optimized anew, whereas the parameter-shift rules are known to perform optimally at fixed shifts.

Finally, we note that Eq. (11) is closely related to the Riemannian gradient on SU(N) [48, 49]. However, instead of a gradient flow on a Lie group, we have defined a flow on the Lie algebra $\mathfrak{su}(N)$, which we retract back to the manifold via the exponential map. This subtle difference induces a different flow from the SU(N) one, as we illustrate in App. C.

Algorithm 1: SU(N) gradients.

Input: $U(\boldsymbol{x})$, ρ, H, $\boldsymbol{\theta}$

Obtain the Jacobian function:

for $l \in (1,\dots,N^2-1)$ **do**

  | $\partial_{x_l} U_l(\boldsymbol{x}) = \partial_{x_l}\mathfrak{Re}[U(\boldsymbol{x})] + i\partial_{x_l}\mathfrak{Im}[U(\boldsymbol{x})]$

end

For each gradient step:

for $l \in (1,\dots,N^2-1)$ **do**

  | $\Omega_l(\boldsymbol{\theta}) \leftarrow U^\dagger(\boldsymbol{\theta})dU_l(\boldsymbol{x})|_{\boldsymbol{\theta}}$

  | $C(t) \leftarrow$

  | $\mathrm{Tr}\Big\{U(\boldsymbol{\theta})e^{t\Omega_l(\boldsymbol{\theta})}\rho e^{-t\Omega_l(\boldsymbol{\theta})}U^\dagger(\boldsymbol{\theta})H\Big\}$

  | $\frac{\partial}{\partial\theta_l}C(\boldsymbol{\theta}) \leftarrow \mathtt{gpsr}(\Omega_l(\boldsymbol{\theta}))$

end

---

4 Comparison with decomposed unitaries

Previous parameterizations of SU(N) unitaries consist of products of single-qubit gates and CNOTs [14, 15, 16, 17, 18, 19]. We refer to this parameterization as *decomposed* $\mathbb{SU}(N)$ gates. On the other hand, Eq. (2) describes a general SU(N) unitary by exponentiating a parameterization of the Lie algebra $\mathfrak{su}(N)$. Here, we investigate the effects of this alternative parameterization.

4.1 Gate speed limit

First, we investigate a speed limit in terms of the gate time. We slightly modify the definition of Eq. (2) for a unitary evolution of the system, $U(\boldsymbol{\theta};t) \in \mathrm{SU}(N)$, to include a time $t \in \mathbb{R}^+$,

$$U(\boldsymbol{\theta};t) = \exp\big\{\bar{A}(\boldsymbol{\theta})t\big\}, \qquad (12)$$

where $\bar{A}(\boldsymbol{\theta}) = A(\boldsymbol{\theta})/\sqrt{\mathrm{Tr}\{A(\boldsymbol{\theta})^\dagger A(\boldsymbol{\theta})\}}$ is a normalized time-independent Hamiltonian (the imaginary unit i is included in $A(\boldsymbol{\theta})$). The normalization of $\bar{A}(\boldsymbol{\theta})$ is equivalent to the normalization of $\boldsymbol{\theta}$ in Euclidean norm, see Lemma 3 in App. F. The normalization of the Hamiltonian (or, equivalently, $\boldsymbol{\theta}$) means that the total path length covered by the evolution is directly proportional to the evolution time t, since we are effectively setting the speed of the evolution to 1.

The Lie group SU(N) can be turned into a Riemannian manifold by equipping it with the Hilbert-Schmidt inner product $g(x,y) = $ $\mathrm{Tr}\big\{x^\dagger y\big\}$. The unitary evolution $U(\boldsymbol{\theta};t)$, parameterized by t, is a one-parameter subgroup that gives the geodesic [48, Theorem III.6] from the identity element at time $t = 0$. Geodesics can be defined as generalizations of straight lines in Euclidean geometry. Using Lemma 4 (App. F), the length of the path after time t is constant for time-independent normalized Hamiltonians with $|\boldsymbol{\theta}| = 1$,

$$L[U(\boldsymbol{\theta};t),t] = \sqrt{N}t. \qquad (13)$$

In general, there is more than one geodesic between two points on the manifold. For example, two points on the Bloch sphere can be connected by rotations about the same axis moving in opposite directions. Using Lemma 5 (App. F), one of these geodesics must be the curve of the minimal path length. Hence, the minimum time to generate the evolution $U(\boldsymbol{\theta};t_g)$ is t_g along the geodesic of the minimal path. For an initial state ρ and final state ρ_f, the Fubini-Study metric is used to find a minimum evolution time

$$t_g = \frac{1}{\sqrt{N}}\arccos\left(\sqrt{\mathrm{Tr}\{\rho\rho_f\}}\right), \qquad (14)$$

giving the Mandelstam-Tamm bound for time-independent normalized Hamiltonians.

In practice, we may only have access to a restricted family of gates within SU(N), for example due to hardware limitations, in which case we require a decomposition of a desired gate in SU(N) into gates from this family. Here we want to compute the additional evolution time required by such a decomposition. The simplest gate decomposition is to break the unitary into two terms, $U(\boldsymbol{\theta};t_g) = U(\boldsymbol{\phi}^{(2)};t_2)U(\boldsymbol{\phi}^{(1)};t_1)$. The parameters $\boldsymbol{\phi}^{(1)}$ and $\boldsymbol{\phi}^{(2)}$ are also normalized Hamiltonians, i.e., they have the norm $|\boldsymbol{\phi}^{(1)}| = |\boldsymbol{\phi}^{(2)}| = 1$. The following theorem shows that using a decomposed circuit over an $\mathbb{SU}(N)$ gate gives an additional evolution time, which corresponds to longer circuit run times.

Theorem 1. *For unitary gates generated by normalized time-independent Hamiltonians, consider a general circuit decomposition of two gates $U(\boldsymbol{\phi}^{(2)};t_2)U(\boldsymbol{\phi}^{(1)};t_1)$. There exists an equivalent evolution with an $\mathbb{SU}(N)$ gate $U(\boldsymbol{\theta};t_g) = U(\boldsymbol{\phi}^{(2)};t_2)U(\boldsymbol{\phi}^{(1)};t_1)$, with evolution time t_g, such that*

$$t_g \leq t_1 + t_2,$$

with equality if $\boldsymbol{\phi}^{(1)} + \boldsymbol{\phi}^{(2)} = \boldsymbol{\theta}$.

The proof of the theorem is in App. F. As expected, a decomposition into two gates gives a longer total evolution time than is possible with an $\mathbb{SU}(N)$ gate due to the normalizations of $\boldsymbol{\phi}^{(1)}$, $\boldsymbol{\phi}^{(2)}$, and $\boldsymbol{\theta}$. A decomposition into more gates would generally lead to an even greater evolution time. A corollary of Theorem 1 is that any circuit with multiple non-commuting layers of gates cannot be optimal in total time.

4.2 Unbiased cost landscapes

An additional advantage of the $\mathbb{SU}(N)$ gate is that it weighs all optimization directions equally. In contrast, a parameterization of SU(N) in terms of a product of gates will create a bias in the parameter space. We illustrate this point with the following example. Consider the decomposed $\mathbb{SU}(2)$ gate $V(\boldsymbol{\theta}) = R_Z(\theta_3)R_Y(\theta_2)R_Z(\theta_1)$ where $R_A(\theta) = \exp\{i\theta A\}$ and $A = X, Y, Z$. This is the ZYZ decomposition. Using similar techniques as in App. F, we can rewrite $V(\boldsymbol{\theta})$ to be parameterized in terms of the Lie algebra:

$$V(\boldsymbol{\theta}) = \exp\{i\boldsymbol{\phi} \cdot \boldsymbol{\sigma}\}, \tag{15}$$

where $\boldsymbol{\sigma} = (X, Y, Z)$ and

$$\boldsymbol{\phi} = \frac{\arccos(\cos(\theta_2)\cos(\theta_1 + \theta_3))}{\sqrt{1 - \cos^2(\theta_2)\cos^2(\theta_1 + \theta_3)}}$$
$$\times \begin{pmatrix} \sin(\theta_2)\sin(\theta_1 - \theta_3) \\ \sin(\theta_2)\cos(\theta_1 - \theta_3) \\ \cos(\theta_2)\sin(\theta_1 + \theta_3) \end{pmatrix}. \tag{16}$$

If we look at the components of $\boldsymbol{\phi}$, we see that the different directions in the Lie algebra are stretched or compressed as a result of the particular choice of parameterization. Consider the normalization $|\theta_1| + |\theta_2| + |\theta_3| = 1$ for the ZYZ decomposition and $|\boldsymbol{\theta}| = 1$ for the $\mathbb{SU}(N)$ gate. With each Hamiltonian term normalized to 1, the prefactor gives the evolution time. These choices of norm give equal total evolution times for the ZYZ decomposition and $\mathbb{SU}(2)$ gate, $T_{\text{ZYZ}} = T_{\text{SU}(N)} = \sqrt{2}$, irrespective of the specific parameters chosen. In Fig. 5, we graphically illustrate the Lie algebra deformation by showing the $\boldsymbol{\phi}$ surface for both the ZYZ decomposition and $\mathbb{SU}(2)$ gate. Note that we have not considered any cost function here; the bias occurs at the level of the parameterization of a specific unitary.

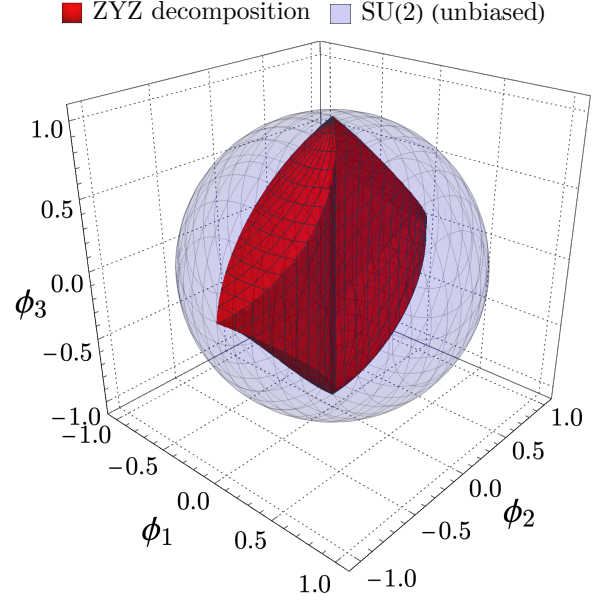The effect of this bias is demonstrated in Fig. 6 for the simplest case of a single-qubit system with

Figure 5: The total unitary evolution for the ZYZ decomposition (red) and the $\mathbb{SU}(2)$ gate (blue) can be expressed in the form $\exp\{i\boldsymbol{\phi} \cdot \boldsymbol{\sigma}\}$. The components $\boldsymbol{\phi} = (\phi_1, \phi_2, \phi_3)$ give the magnitude of the respective basis generators $\boldsymbol{\sigma} = (X, Y, Z)$. The original parameterization in $\boldsymbol{\theta}$ with norm $|\theta_1| + |\theta_2| + |\theta_3| = 1$ gives a surface of possible values of $\boldsymbol{\phi}$ and therefore possible unitary evolutions. The $\mathbb{SU}(2)$ gate (blue) is unbiased because its parameterization gives the correspondence $\boldsymbol{\theta} = \boldsymbol{\phi}$ with normalization $\phi_1^2 + \phi_2^2 + \phi_3^2 = 1$. The unitary evolution for the ZYZ decomposition (red) is biased because the surface in the $\boldsymbol{\phi}$ coordinates does not maintain an equal magnitude in all directions.

an $\mathbb{SU}(2)$ gate. The optimal parameters of the circuit are those that produce the state that gives the minimum of the cost function $C(\boldsymbol{\theta}) = -\langle Y \rangle$ (green star). We consider various initial parameters acting on the reference state $\rho = |0\rangle\langle 0|$. The corresponding training paths are shown for each initial parameter vector. The training paths for the decomposed ZYZ circuit are depicted in Fig. 6(a). As the initial parameter $\boldsymbol{\theta}_0$ acting on the reference state ρ (purple dots) moves closer to an unstable equilibrium point (orange diamond) the training path becomes increasingly suboptimal. At the unstable equilibrium the only gradient information is directly away from the instability rather than providing information about the direction towards the global minimum. This behavior is further illustrated by the gradient vector field on the Bloch sphere in Fig. 6(c). For the $\mathbb{SU}(N)$ gate, we see in Fig. 6(b) that the optimization trajectories follow a direct path to the minimum.
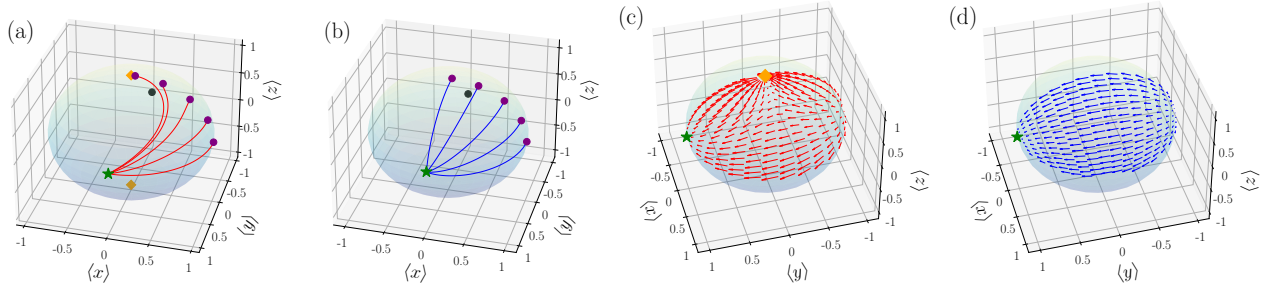
Figure 6: Comparison of the update of circuit parameters from various initial parameters acting on the initial state $\rho = |0\rangle\langle 0|$. The training paths are depicted on the Bloch sphere for: (a) parameterized single-qubit rotations for the ZYZ ansatz; and (b) using the $\mathbb{SU}(N)$ gate. The purple dots represent initial states generated by applying $U(\boldsymbol{\theta}_0)$ with $\boldsymbol{\theta}_0 = (0, a, 0)$ where $a \in \left\{ \frac{\pi}{64}, \frac{\pi}{8}, \frac{2\pi}{8}, \frac{3\pi}{8}, \frac{\pi}{2} \right\}$ to ρ. Note that for this choice of initial parameters, $U(\boldsymbol{\theta}_0) = V(\boldsymbol{\theta}_0)$. The objective function is $C(\boldsymbol{\theta}) = -\langle Y \rangle$, giving the target final state at the green star—the state that gives the global minimum of $C(\boldsymbol{\theta})$. The unstable equilibrium points are given by orange diamonds, at $(0, 0, 1)$ and $(0, 0, -1)$, and the black point is at the maximum of the cost function, $(0, 1, 0)$. (c) shows the gradient vector field of the decomposed ZYZ ansatz. The vector field for the $\mathbb{SU}(2)$ gate, shown in (d), coincides with the geodesic flow towards the target final state at all points which satisfies the gate speed limit.

4.3 Numerical experiment

To investigate the effect on the performance of a typical optimization, we study how an $\mathbb{SU}(N)$ gate compares with a decomposed gate in a larger circuit. In Fig. 7 we provide a non-trivial example, where we incorporate our gates into a circuit and show that it performs better than a decomposed $\mathbb{SU}(4)$ gate on a set of random problem instances. We show the individual optimization trajectories in Fig. 8 which illustrate the faster optimization of $\mathbb{SU}(N)$ gates compared to decomposed gates. Like for the examples in Fig. 3 and Fig. 4, we assume that there is no gate or measurement noise. Additionally, we assume that we can always implement the gate generated by $\Omega_l(\boldsymbol{\theta})$, and have control over all Pauli operators G_m. In practice, we typically only have access to a fixed set of generators $\text{span}(\{G_m\}) < \text{span}(\mathfrak{su}(N))$. If this is the case, then we require a decomposition of $\exp\{t\Omega_l(\boldsymbol{\theta})\}$ in terms of the available operations on the device [14, 19]. All numerical results were obtained with `PennyLane` [50], and the $\mathbb{SU}(N)$ gates can be accessed via the `qml.SpecialUnitary` class. Although we do not explore this here, one could make use of sparse optimization methods such as stochastic optimization [51, 52] and frugal optimization [53] for the GPSR subroutine in our algorithm.

5 Resource estimation

To obtain the partial derivative in Eq. (11) in practice we need to estimate the gradient of a circuit that contains a gate generated by $\Omega_l(\boldsymbol{\theta})$. As noted in recent works on GPSR rules [8, 7, 9], the computational cost of estimating this gradient is related to the spectral gaps of $\Omega_l(\boldsymbol{\theta})$. In particular, if $\{\lambda_j\}$ is the set of (possibly degenerate) eigenvalues of $\Omega_l(\boldsymbol{\theta})$, we define the set of unique spectral gaps as $\Gamma = \{|\lambda_j - \lambda_{j'}|\}$ where $j' > j$. Note that for d distinct eigenvalues, the number of unique spectral gaps R is at most $R \leq d(d-1)/2$. The total number of parameter-shifted circuits is then $2R$ for a single partial derivative $\partial_{\theta_l} C(\boldsymbol{\theta})$

Depending on the generator $\Omega_l(\boldsymbol{\theta})$, this complexity can be improved. For instance, in [7], a Cartan decomposition is used to improve the number of circuits required from polynomial to linear or even logarithmic in N. Additionally, in [8], the different costs for first- and second-order gradients are determined for specific varational quantum algorithms like QAOA [54] and RotoSolve [55, 56, 57, 58]. Finally, in [9], the computational cost of a variety of different gates is investigated in detail and the variance across the parameter regime is studied.

Instead of focusing on specific instances of the generator $\Omega_l(\boldsymbol{\theta})$, we make a more general observation about the computational complexity of parameter-shift gradient rules. In general, $\Omega_l(\boldsymbol{\theta})$ has full support on $\mathfrak{su}(N)$, since the consecutive
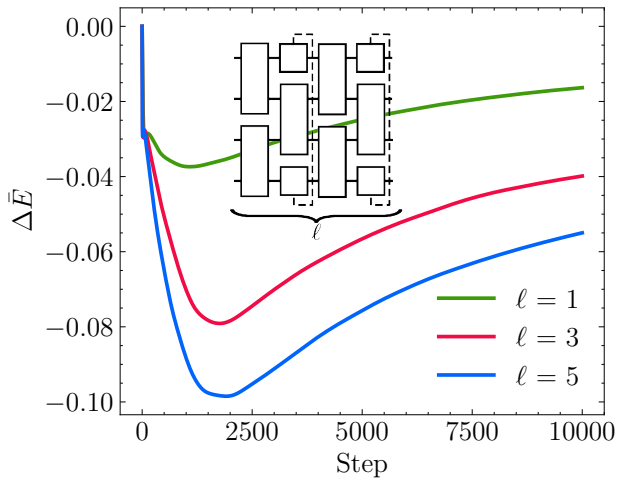
Figure 7: Comparison of decomposed gates versus $\mathbb{SU}(N)$ gates in brick-layer circuits for random 10-qubit Hamiltonians and various depths. We consider the brick-layer circuit indicated with $\ell = 2$ in the inset, with general two-qubit gates acting on the even and odd qubits in each layer. The decomposed gate is the SU(4) parameterization of [16], which is optimal in the number of CNOTs required. For each instance, we sample a Hamiltonian from the Gaussian unitary ensemble and minimize the cost in Eq. (9) via standard gradient descent. We show the difference of the relative errors in energy $\bar{E} = (E - E_{\min})/(E_{\max} - E_{\min})$ between the decomposed gates and the $\mathbb{SU}(N)$ gates, that is $\Delta\bar{E} = \bar{E}_{\mathrm{SU}(N)} - \bar{E}_{\mathrm{Decomp}}$. The plotted lines are the mean $\bar{E}$, averaged over 50 random Hamiltonians for each circuit depth ℓ. We see that for all depths $\Delta\bar{E} < 0$ at all points during the optimization, hence the brick-layer circuit with the $\mathbb{SU}(N)$ gates outperforms the circuit where the two-qubit interactions are parametrized as a gate composition.

applications of $\mathrm{ad}_{A(\boldsymbol{\theta})}$ in Eq. (3) typically generate all of $\mathfrak{su}(N)$ [59]. However, for specific choices of $A(\boldsymbol{\theta})$, the application of $\mathrm{ad}_{A(\boldsymbol{\theta})^p}$ to $\partial_{\theta_l} A(\boldsymbol{\theta})$ closes to form a subalgebra, called the dynamical Lie algebra of $A(\boldsymbol{\theta})$, that is contained in $\mathfrak{su}(N)$. These algebras are well-known in the context of quantum optimal control [60, 61], and have recently been studied in the context of variational quantum algorithms [62, 63]. We define the dynamical Lie algebra (DLA) $\mathcal{L}(A(\boldsymbol{\theta}))$ as the subalgebra formed under the closure of the non-zero terms in $A(\boldsymbol{\theta})$ under the commutator. Ignoring global phases, this will always result in a subalgebra $\mathcal{L}(A(\boldsymbol{\theta})) \subseteq \mathfrak{su}(N)$. For example, given $A(\boldsymbol{\theta}) = i(aX + bY), \forall a, b \in \mathbb{R}$, we have $\mathcal{L}(A(\boldsymbol{\theta})) = \mathrm{span}\{iX, iY, iZ\}$, since $\mathrm{ad}_X(Y) = [X, Y] = iZ$ and successive commutators generate no new contributions. Note that for this example the DLA
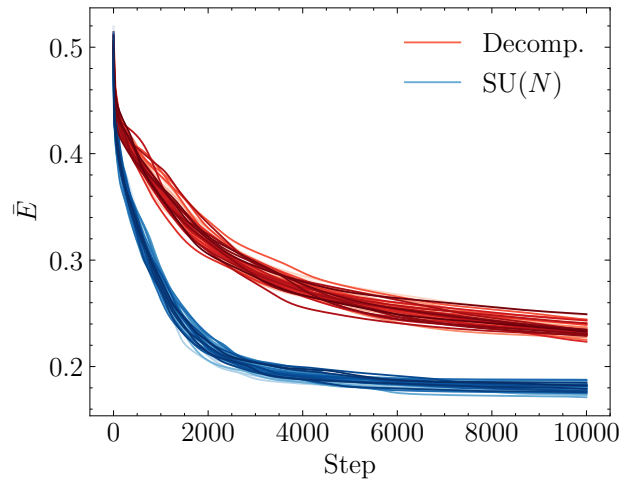

Figure 8: Trajectories from the optimizations in Fig. 7 for 50 random 10-qubit Hamiltonians sampled from the Gaussian unitary ensemble and an $\ell = 5$ brick-layer circuit of 2-qubit building blocks. We compare the relative error energy (see Fig. 7 for the definition of $\bar{E}$) when using a standard gate composition to that when using $\mathbb{SU}(4)$ gates as building blocks. The optimization is performed with vanilla gradient descent using a learning rate of $\eta = 10^{-3}$. The $\mathbb{SU}(4)$ gate consistently leads to faster optimization and better approximations of the ground state energy throughout all 10^5 optimization steps.

equals the full Lie algebra $\mathfrak{su}(2)$. Explicit constructions of DLAs that span $\mathfrak{so}(N)$ and $\mathfrak{sp}(N)$ are given in [64]. In a more recent work, the DLAs of several typical quantum many-body Hamiltonians are studied and their properties are used to prepare efficient time-evolution circuits [65]. In one dimension, the DLAs that are generated by Pauli strings have recently been classified [66].

Interestingly, if the DLA is maximal, i.e., there exists no smaller non-trivial subalgebra within $\mathcal{L}(A(\boldsymbol{\theta}))$, then the *roots* of the Lie algebra can be related directly to the computational cost of estimating the gradients in Eq. (11). We formally establish this connection with the following theorem:

Theorem 2. *The number of unique spectral gaps R of $\Omega_l(\boldsymbol{\theta})$ is upper bounded by the number of roots $|\Phi|$ of any maximal semi-simple DLA,*

$$R \leq |\Phi|/2. \tag{17}$$

We provide the proof of Theorem 2 in App. G. We make use of the fact that any semisimple Lie algebra can be written as a direct sum of its weight spaces, which can be identified with

its root system [67]. The number of roots $|\Phi|$ can then be used to bound the total number of unique spectral gaps of $\Omega_l(\boldsymbol{\theta})$. We can thus use Theorem 2 to assess the run time of Algorithm 1. We give several examples of $\mathbb{SU}(N)$ gates in App. G together with the corresponding values of R. Depending on the physical system or hardware that we are working with, we have to choose a representation for $\mathfrak{su}(N)$, which is a map $\mathfrak{su}(N) \rightarrow \mathfrak{gl}(N, \mathbb{C})$. In Eq. (1) we chose this representation to be the tensor product of the fundamental representation, i.e., Pauli monomials. Note however, that Eq. (11) and Theorem 2 hold for any irreducible representation of $\mathfrak{su}(N)$.

Additionally, by connecting the spectral gaps to the root system of the DLA, we can make use of a beautiful result in representation theory: the classification of all maximal subalgebras of the classical Lie algebras [68]. Each root system can be uniquely identified with a particular subalgebra of a Lie algebra and it can be shown that there exist a finite number of root systems. Since a DLA is a subalgebra of $\mathfrak{su}(N)$, we can identify all possible DLAs and by extension all possible families of $\mathbb{SU}(N)$ gates. We provide examples of this procedure in App. G.

6 Conclusion

We have proposed an alternative parameterization of general $\mathbb{SU}(N)$ gates and a method of optimizing these gates in prototypical variational quantum algorithms. We have shown that our gates are more powerful in toy example settings, and motivated why we believe this is the case based on quantum speed-limit arguments. A natural extension of our work would be to test our method in experimental settings, both on gate-based quantum computers or quantum simulators [69, 70, 71]. With regards to the latter, several methods have been investigated that could provide pulse-level optimization of energy cost functions [72, 73]. This would obviate the need for a gate-based model of quantum computing to prepare specific states on quantum hardware. Instead, we work on the Hamiltonian level and control the system directly. Our algorithm could be applied to this setting as well, since we're effectively learning the parameters of some fixed Hamiltonian.

We have shown that the $\mathbb{SU}(N)$ gate in a circuit outperforms a decomposed gate. The number of parameters in our proposed gate equals $4^{N_{\text{qubits}}}$, hence $\mathbb{SU}(N)$ gates acting on a large number of qubits will be impractical. Additionally, it is not clear for which problems one would rather have a deeper circuit with simple gates as opposed to a shallow circuit with more powerful gates. This also begs another question: will our gates suffer from barren plateaus [74]? It is likely that a circuit of 2-qubit $\mathbb{SU}(N)$ gates that has linear depth in N will lead to a circuit that forms an approximate 2-design, which will suffer from vanishing gradients. However, appropriate choices of the generators $A(\boldsymbol{\theta})$ of our gate could keep the circuit in a polynomially scaling DLA of the entire circuit, which can avoid barren plateaus [62, 63]. Additionally, we can consider parameter initialization strategies that can improve the optimization [75, 76].

Finally, we believe that the connections between variational quantum circuits and representation theory merit further investigation. We connected the classification of all $\mathbb{SU}(N)$ gates with the classification of semisimple Lie algebras. However, this could possibly be extended to a classification of all variational quantum circuits based on the DLA of an ansatz. It seems that the tools to provide such a classification are available and could provide one with a method to assess the trainability and expressivity of variational circuits without explicitly referring to specific ansätze.

7 Acknowledgements

References

[1] M. Cerezo, Andrew Arrasmith, Ryan Babbush, Simon C. Benjamin, Suguru Endo, Keisuke Fujii, Jarrod R. McClean, Kosuke Mitarai, Xiao Yuan, Lukasz Cincio, and Patrick J. Coles. "Variational quantum algorithms". Nature Reviews Physics **3**, 625–644 (2021).

[2] Jules Tilly, Hongxiang Chen, Shuxiang Cao, Dario Picozzi, Kanav Setia, Ying Li, Edward Grant, Leonard Wossnig, Ivan Rungger, George H. Booth, and Jonathan Tennyson. "The Variational Quantum Eigensolver: A review of methods and best practices". Physics Reports **986**, 1–128 (2022).

[3] Jun Li, Xiaodong Yang, Xinhua Peng, and Chang-Pu Sun. "Hybrid Quantum-Classical Approach to Quantum Optimal Control". Phys. Rev. Lett. **118**, 150503 (2017).

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

[5] Maria Schuld, Ville Bergholm, Christian Gogolin, Josh Izaac, and Nathan Killoran. "Evaluating analytic gradients on quantum hardware". Phys. Rev. A **99**, 032331 (2019).

[6] Gavin E. Crooks. "Gradients of parameterized quantum gates using the parameter-shift rule and gate decomposition" (2019) arXiv:1905.13311.

[7] Artur F. Izmaylov, Robert A. Lang, and Tzu-Ching Yen. "Analytic gradients in variational quantum algorithms: Algebraic extensions of the parameter-shift rule to general unitary transformations". Phys. Rev. A **104**, 062443 (2021).

[8] David Wierichs, Josh Izaac, Cody Wang, and Cedric Yen-Yu Lin. "General parameter-shift rules for quantum gradients". Quantum **6**, 677 (2022).

[9] Oleksandr Kyriienko and Vincent E. Elfving. "Generalized quantum circuit differentiation rules". Phys. Rev. A **104**, 052417 (2021).

[10] Dirk Oliver Theis. ""Proper" Shift Rules for Derivatives of Perturbed-Parametric Quantum Evolutions". Quantum **7**, 1052 (2023).

[11] Lucas Slattery, Benjamin Villalonga, and Bryan K. Clark. "Unitary block optimization for variational quantum algorithms". Phys. Rev. Research **4**, 023072 (2022).

[12] Jin-Guo Liu, Yi-Hong Zhang, Yuan Wan, and Lei Wang. "Variational quantum eigensolver with fewer qubits". Phys. Rev. Research **1**, 023025 (2019).

[13] Abhinav Kandala, Antonio Mezzacapo, Kristan Temme, Maika Takita, Markus Brink, Jerry M. Chow, and Jay M. Gambetta. "Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets". Nature **549**, 242–246 (2017).

[14] Navin Khaneja and Steffen J. Glaser. "Cartan decomposition of $SU(2^n)$ and control of spin systems". Chemical Physics **267**, 11–23 (2001).

[15] Barbara Kraus and Juan I Cirac. "Optimal creation of entanglement using a two-qubit gate". Physical Review A **63**, 062309 (2001).

[16] Farrokh Vatan and Colin Williams. "Optimal quantum circuits for general two-qubit gates". Phys. Rev. A **69**, 032315 (2004).

[17] Farrokh Vatan and Colin P Williams. "Realization of a general three-qubit quantum gate" (2004). arXiv:quant-ph/0401178.

[18] Juha J. Vartiainen, Mikko Möttönen, and Martti M. Salomaa. "Efficient Decomposition of Quantum Gates". Phys. Rev. Lett. **92**, 177902 (2004).

[19] Domenico D'Alessandro and Raffaele Romano. "Decompositions of unitary evolutions and entanglement dynamics of bipartite quantum systems". Journal of Mathematical Physics **47**, 082109 (2006).

[20] Alwin Zulehner and Robert Wille. "Compiling SU(4) Quantum Circuits to IBM QX Architectures". In Proceedings of the 24th Asia and South Pacific Design Automation Conference. Page 185–190. ASPDAC '19New

York, NY, USA (2019). Association for Computing Machinery.

[21] B. Foxen, C. Neill, A. Dunsworth, P. Roushan, B. Chiaro, A. Megrant, J. Kelly, Zijun Chen, K. Satzinger, R. Barends, F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, S. Boixo, D. Buell, B. Burkett, Yu Chen, R. Collins, E. Farhi, A. Fowler, C. Gidney, M. Giustina, R. Graff, M. Harrigan, T. Huang, S. V. Isakov, E. Jeffrey, Z. Jiang, D. Kafri, K. Kechedzhi, P. Klimov, A. Korotkov, F. Kostritsa, D. Landhuis, E. Lucero, J. McClean, M. McEwen, X. Mi, M. Mohseni, J. Y. Mutus, O. Naaman, M. Neeley, M. Niu, A. Petukhov, C. Quintana, N. Rubin, D. Sank, V. Smelyanskiy, A. Vainsencher, T. C. White, Z. Yao, P. Yeh, A. Zalcman, H. Neven, and J. M. Martinis. "Demonstrating a Continuous Set of Two-Qubit Gates for Near-Term Quantum Algorithms". Phys. Rev. Lett. **125**, 120504 (2020).

[22] E Groeneveld. "A reparameterization to improve numerical optimization in multivariate REML (co) variance component estimation". Genetics Selection Evolution **26**, 537–545 (1994).

[23] Tapani Raiko, Harri Valpola, and Yann Lecun. "Deep learning made easier by linear transformations in perceptrons". In Neil D. Lawrence and Mark Girolami, editors, Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics. Volume 22 of Proceedings of Machine Learning Research, pages 924–932. La Palma, Canary Islands (2012). PMLR. url: https://proceedings.mlr.press/v22/raiko12.html.

[24] Sergey Ioffe and Christian Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift". In International conference on machine learning. Pages 448–456. PMLR (2015).

[25] Tim Salimans and Durk P Kingma. "Weight normalization: A simple reparameterization to accelerate training of deep neural networks". In Advances in neural information processing systems. Volume 29. (2016).

[26] Robert Price. "A useful theorem for nonlinear devices having Gaussian inputs". IRE Transactions on Information Theory **4**, 69–72 (1958).

[27] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. "Stochastic backpropagation and approximate inference in deep generative models". In Eric P. Xing and Tony Jebara, editors, Proceedings of the 31st International Conference on Machine Learning. Volume 32 of Proceedings of Machine Learning Research, pages 1278–1286. Bejing, China (2014). PMLR. url: https://proceedings.mlr.press/v32/rezende14.html.

[28] Diederik P. Kingma and Max Welling. "Auto-Encoding Variational Bayes". In Yoshua Bengio and Yann LeCun, editors, 2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings. (2014). url: http://arxiv.org/abs/1312.6114.

[29] Brian C Hall. "Lie groups, Lie algebras, and representations". Springer. (2013). 2nd edition.

[30] William Fulton and Joe Harris. "Representation theory: a first course". Volume 129. Springer Science & Business Media. (2013).

[31] W. Rossmann. "Lie Groups: An Introduction Through Linear Groups". Oxford graduate texts in mathematics. Oxford University Press. (2002). 5th edition.

[32] Jean-Pierre Serre. "Lie algebras and Lie groups: 1964 lectures given at Harvard University". Springer. (2009).

[33] Norbert Schuch and Jens Siewert. "Natural two-qubit gate for quantum computation using the XY interaction". Phys. Rev. A **67**, 032301 (2003).

[34] T. P. Orlando, J. E. Mooij, Lin Tian, Caspar H. van der Wal, L. S. Levitov, Seth Lloyd, and J. J. Mazo. "Superconducting persistent-current qubit". Phys. Rev. B **60**, 15398–15413 (1999).

[35] B. E. Kane. "A silicon-based nuclear spin quantum computer". Nature **393**, 133–137 (1998).

[36] A. Imamoḡlu, D. D. Awschalom, G. Burkard, D. P. DiVincenzo, D. Loss,

M. Sherwin, and A. Small. "Quantum information processing using quantum dot spins and cavity qed". Phys. Rev. Lett. **83**, 4204–4207 (1999).

[37] Jiaqi Leng, Yuxiang Peng, Yi-Ling Qiao, Ming Lin, and Xiaodi Wu. "Differentiable Analog Quantum Computing for Optimization and Control" (2022). arXiv:2210.15812.

[38] R. M. Wilcox. "Exponential Operators and Parameter Differentiation in Quantum Physics". Journal of Mathematical Physics **8**, 962–982 (1967). arXiv:https://doi.org/10.1063/1.1705306.

[39] E. T. Whittaker. "XVIII.—On the Functions which are represented by the Expansions of the Interpolation-Theory". Proceedings of the Royal Society of Edinburgh **35**, 181–194 (1915).

[40] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang (2018). code: google/jax.

[41] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. "Pytorch: An imperative style, high-performance deep learning library". In Advances in neural information processing systems. Volume 32. (2019).

[42] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng (2015). code: https://www.tensorflow.org/.

[43] A JAX implementation of the matrix exponential that can be differentiated via automatic differentiation: https://jax.readthedocs.io/en/latest/_autosummary/jax.scipy.linalg.expm.html.

[44] Awad H Al-Mohy and Nicholas J Higham. "A new scaling and squaring algorithm for the matrix exponential". SIAM Journal on Matrix Analysis and Applications **31**, 970–989 (2010).

[45] Leonardo Banchi and Gavin E. Crooks. "Measuring Analytic Gradients of General Quantum Evolution with the Stochastic Parameter Shift Rule". Quantum **5**, 386 (2021).

[46] Lennart Bittel, Jens Watty, and Martin Kliesch. "Fast gradient estimation for variational quantum algorithms" (2022). arXiv:2210.06484.

[47] Roeland Wiersema, Dylan Lewis, David Wierichs, Juan Carrasquilla, and Nathan Killoran (2023). code: dwierichs/Here-comes-the-SUN.

[48] Thomas Schulte-Herbrüggen, Steffen j. Glaser, Gunther Dirr, and Uwe Helmke. "Gradient Flows for Optimization in Quantum Information and Quantum Dynamics: Foundations and Applications". Reviews in Mathematical Physics **22**, 597–667 (2010).

[49] Roeland Wiersema and Nathan Killoran. "Optimizing quantum circuits with riemannian gradient flow" (2023).

[50] Ville Bergholm, Josh Izaac, Maria Schuld, Christian Gogolin, M Sohaib Alam, Shahnawaz Ahmed, Juan Miguel Arrazola, Carsten Blank, Alain Delgado, Soran Jahangiri, et al. "Pennylane: Automatic differentiation of hybrid quantum-classical computations" (2018). arXiv:1811.04968.

[51] Ryan Sweke, Frederik Wilde, Johannes Meyer, Maria Schuld, Paul K. Faehrmann, Barthélémy Meynard-Piganeau, and Jens Eisert. "Stochastic gradient descent for hybrid quantum-classical optimization". Quantum **4**, 314 (2020).

[52] Aram W. Harrow and John C. Napp. "Low-Depth Gradient Measurements Can Improve Convergence in Variational Hybrid

Quantum-Classical Algorithms". Phys. Rev. Lett. **126**, 140502 (2021).

[53] Andrew Arrasmith, Lukasz Cincio, Rolando D Somma, and Patrick J Coles. "Operator sampling for shot-frugal optimization in variational algorithms" (2020). arXiv:2004.06252.

[54] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. "A quantum approximate optimization algorithm" (2014). arXiv:1411.4028.

[55] Javier Gil Vidal and Dirk Oliver Theis. "Calculus on parameterized quantum circuits" (2018). arXiv:1812.06323.

[56] Robert M Parrish, Joseph T Iosue, Asier Ozaeta, and Peter L McMahon. "A Jacobi diagonalization and Anderson acceleration algorithm for variational quantum algorithm parameter optimization" (2019). arXiv:1904.03206.

[57] Ken M. Nakanishi, Keisuke Fujii, and Synge Todo. "Sequential minimal optimization for quantum-classical hybrid algorithms". Phys. Rev. Res. **2**, 043158 (2020).

[58] Mateusz Ostaszewski, Edward Grant, and Marcello Benedetti. "Structure optimization for parameterized quantum circuits". Quantum **5**, 391 (2021).

[59] Seth Lloyd. "Universal quantum simulators". Science **273**, 1073–1078 (1996).

[60] F. Albertini and D. D'Alessandro. "Notions of controllability for quantum mechanical systems". In Proceedings of the 40th IEEE Conference on Decision and Control (Cat. No.01CH37228). Volume 2, pages 1589–1594 vol.2. (2001).

[61] Domenico d'Alessandro. "Introduction to quantum control and dynamics". Chapman and hall/CRC. (2021). 2nd edition.

[62] Martin Larocca, Piotr Czarnik, Kunal Sharma, Gopikrishnan Muraleedharan, Patrick J. Coles, and M. Cerezo. "Diagnosing Barren Plateaus with Tools from Quantum Optimal Control". Quantum **6**, 824 (2022).

[63] Martín Larocca, Nathan Ju, Diego García-Martín, Patrick J. Coles, and Marco Cerezo. "Theory of overparametrization in quantum

neural networks". Nature Computational Science **3**, 542–551 (2023).

[64] S G Schirmer, I C H Pullen, and A I Solomon. "Identification of dynamical Lie algebras for finite-level quantum control systems". Journal of Physics A: Mathematical and General **35**, 2327 (2002).

[65] Efekan Kökcü, Thomas Steckmann, Yan Wang, J. K. Freericks, Eugene F. Dumitrescu, and Alexander F. Kemper. "Fixed depth hamiltonian simulation via cartan decomposition". Phys. Rev. Lett. **129**, 070501 (2022).

[66] Roeland Wiersema, Efekan Kökcü, Alexander F Kemper, and Bojko N Bakalov. "Classification of dynamical lie algebras for translation-invariant 2-local spin systems in one dimension" (2023). arXiv:2203.05690.

[67] Jean-Pierre Serre. "Complex semisimple Lie algebras". Springer Science & Business Media. (2000). 1st edition.

[68] Eugene Borisovich Dynkin. "American Mathematical Society Translations: Five Papers on Algebra and Group Theory". American Mathematical Society. (1957).

[69] I. M. Georgescu, S. Ashhab, and Franco Nori. "Quantum simulation". Rev. Mod. Phys. **86**, 153–185 (2014).

[70] Sepehr Ebadi, Tout T Wang, Harry Levine, Alexander Keesling, Giulia Semeghini, Ahmed Omran, Dolev Bluvstein, Rhine Samajdar, Hannes Pichler, Wen Wei Ho, et al. "Quantum phases of matter on a 256-atom programmable quantum simulator". Nature **595**, 227–232 (2021).

[71] P. Scholl, H. J. Williams, G. Bornet, F. Wallner, D. Barredo, L. Henriet, A. Signoles, C. Hainaut, T. Franz, S. Geier, A. Tebben, A. Salzinger, G. Zürn, T. Lahaye, M. Weidemüller, and A. Browaeys. "Microwave Engineering of Programmable XXZ Hamiltonians in Arrays of Rydberg Atoms". PRX Quantum **3**, 020303 (2022).

[72] Mohannad Ibrahim, Hamed Mohammadbagherpoor, Cynthia Rios, Nicholas T Bronn, and Gregory T Byrd. "Pulse-Level Optimization of Parameterized Quantum Circuits for Variational Quantum Algorithms" (2022). arXiv:2211.00350.

[73] Oinam Romesh Meitei, Bryan T. Gard, George S. Barron, David P. Pappas, Sophia E. Economou, Edwin Barnes, and Nicholas J. Mayhall. "Gate-free state preparation for fast variational quantum eigensolver simulations". npj Quantum Information **7**, 155 (2021).

[74] Jarrod R McClean, Sergio Boixo, Vadim N Smelyanskiy, Ryan Babbush, and Hartmut Neven. "Barren plateaus in quantum neural network training landscapes". Nature communications **9**, 1–6 (2018).

[75] Edward Grant, Leonard Wossnig, Mateusz Ostaszewski, and Marcello Benedetti. "An initialization strategy for addressing barren plateaus in parametrized quantum circuits". Quantum **3**, 214 (2019).

[76] Andrea Skolik, Jarrod R McClean, Masoud Mohseni, Patrick van der Smagt, and Martin Leib. "Layerwise learning for quantum neural networks". Quantum Machine Intelligence **3**, 1–11 (2021).

[77] Rüdiger Achilles and Andrea Bonfiglioli. "The early proofs of the theorem of Campbell, Baker, Hausdorff, and Dynkin". Archive for History of Exact Sciences **66**, 295–358 (2012).

[78] Mario Lezcano-Casado and David Martınez-Rubio. "Cheap orthogonal constraints in neural networks: A simple parametrization of the orthogonal and unitary group". In International Conference on Machine Learning. Pages 3794–3803. PMLR (2019).

[79] Andrea Mari, Thomas R. Bromley, and Nathan Killoran. "Estimating the gradient and higher-order derivatives on quantum hardware". Phys. Rev. A **103**, 012405 (2021).

[80] Benjamin Russell and Susan Stepney. "Geometric Methods for Analysing Quantum Speed Limits: Time-Dependent Controlled Quantum Systems with Constrained Control Functions". In Giancarlo Mauri, Alberto Dennunzio, Luca Manzoni, and Antonio E. Porreca, editors, Unconventional Computation and Natural Computation. Pages 198–208. Lecture Notes in Computer ScienceBerlin, Heidelberg (2013). Springer.

[81] Andreas Arvanitogeōrgos. "An introduction to Lie groups and the geometry of homogeneous spaces". Volume 22. American Mathematical Soc. (2003).

[82] S Helgason. "Differential geometry, lie groups, and symmetric spaces". American Mathematical Soc. (1978).

[83] James E Humphreys. "Introduction to Lie algebras and representation theory". Volume 9. Springer Science & Business Media. (2012).

A $\mathrm{SU}(N)$ and its Lie algebra

Consider the special unitary Lie group $\mathrm{SU}(N)$:

$$\mathrm{SU}(N) = \{X \in \mathbb{C}^{N \times N} | X^\dagger X = I, \det\{X\} = 1\}. \tag{A1}$$

In general, quantum gates belong to the unitary group $\mathrm{U}(N)$, which drops the determinant 1 condition and thus allows for an additional global phase. Restricting ourselves to $\mathrm{SU}(N)$ therefore is physically equivalent. Consider a curve $X(t) : \mathbb{R} \to \mathrm{SU}(N)$, $t \in (-1, 1)$ such that $X(0) = I$ and $d/dt X(t)|_{t=0} \equiv \dot{X}(0) = \Omega$. If we differentiate the unitarity condition with respect to t, we obtain

$$\frac{d}{dt}(X^\dagger(t)X(t))|_{t=0} = 0, \tag{A2}$$

$$\dot{X}^\dagger(t)X(t) + X^\dagger(t)\dot{X}(t)|_{t=0} = 0 \tag{A3}$$

$$\Omega^\dagger + \Omega = 0, \tag{A4}$$

and so we find that $\Omega^\dagger = -\Omega$, i.e., Ω is a skew-Hermitian matrix. The matrices Ω from all such curves are elements of the Lie algebra

$$\mathfrak{su}(N) = \{\Omega \in \mathbb{C}^{N \times N} | \Omega^\dagger = -\Omega, \mathrm{Tr}\{\Omega\} = 0\}. \tag{A5}$$

The second condition, $\mathrm{Tr}\{\Omega\} = 0$, can be found by realizing that the Lie algebra is connected to a Lie group via the exponential map: $e^{tX} \in \mathrm{SU}(N)$, $\forall t \in \mathbb{R}$ and $X \in \mathfrak{su}(N)$ and so $1 = \det\{e^X\} = e^{\mathrm{Tr}\{X\}}$. Instead of considering a curve going through the identity at $t = 0$, we can consider a curve going through a point U with directional derivative Ω. All curves $X(t)$ with directional derivative Ω at the point U form an equivalence class. The set of these equivalence classes forms a vector space called the tangent space $T_U \mathrm{SU}(N)$ at the point U, and in particular we may identify $T_I \mathrm{SU}(N) = \mathfrak{su}(N)$. The group product L_U, i.e., left multiplication by an element U, is a local homeomorphism, which induces a linear map between two tangent spaces: $d(L_U)|_V : T_V \mathrm{SU}(N) \to T_{L_U(V)} \mathrm{SU}(N)$ where $U, V \in \mathrm{SU}(N)$. $d(L_U)$ is called the differential of L_U. If we take $V = I$ to be the identity, then we see that we can move from the tangent space at the identity to the tangent space at any point in $\mathrm{SU}(N)$ by left multiplication of the group $d(L_U)|_I : \mathfrak{su}(N) \to T_U \mathrm{SU}(N)$. Hence the tangent space at U is given by

$$T_U \mathrm{SU}(N) = \{U\Omega | \Omega \in \mathfrak{su}(N)\}. \tag{A6}$$

The above derivation can provide us with a different understanding of a what the Lie algebra is. Starting from the group $\mathrm{SU}(N)$, we can consider all left-invariant vector fields X on $\mathrm{SU}(N)$,

$$d(L_U)|_V(X_V) = X|_{U \cdot V} \tag{A7}$$

where $X \in \mathfrak{X}(\mathrm{SU}(N))$, which is the space of vector fields on $\mathrm{SU}(N)$. The left-invariant vector fields form a vector and since they are closed under the Lie bracket $[X, Y]$, they define a Lie algebra.

B The exponential map and its gradient

The following is due to [31, Chapter 1, Theorem 5]. Let G be a matrix Lie group $G \in \mathrm{GL}(N, \mathbb{C})$ with a corresponding Lie algebra $\mathfrak{g}$. Define conjugation by $h \in G$ to be the transformation $c_h : G \to G$ given by $c_h(g) = hgh^{-1}$. Note that c is an (inner) automorphism of G, since it is a isomorphism from G onto itself. Let $\exp : \mathfrak{g} \to G$ be the exponential map from the Lie algebra to the group. Taking $X \in \mathfrak{g}$ and $t \in \mathbb{R}$, the differential of the conjugation map at the identity is

$$d(c_h)(X) = \frac{d}{dt}(he^{tX}h^{-1})\Big|_{t=0} = hXh^{-1}, \tag{B1}$$

which maps an element of the Lie algebra to another element of the Lie algebra. This map is called the adjoint representation $\mathrm{Ad}_h : \mathfrak{g} \to \mathfrak{g}$ and takes $X \mapsto hXh^{-1}$. Given $X, Y \in \mathfrak{g}$, we compute

$$\frac{d}{dt}\mathrm{Ad}_{e^{tX}} Y \bigg|_{t=0} = \frac{d}{dt}e^{tX}Ye^{-tX}\bigg|_{t=0} \tag{B2}$$

$$= XY - YX \tag{B3}$$

$$= [X, Y] \tag{B4}$$

$$= \mathrm{ad}_X Y. \tag{B5}$$

The operator $\mathrm{ad} : \mathfrak{g} \times \mathfrak{g} \to \mathfrak{g}$ is the Lie bracket on $\mathfrak{g}$, which for our purposes will be the standard commutator. It now follows that

$$\frac{d}{dt}\mathrm{Ad}_{e^{tX}} Y = \frac{d}{dt}\left(e^{tX}Ye^{-tX}\right) \tag{B6}$$

$$= Xe^{tX}Ye^{-tX} + e^{tX}Ye^{-tX}(-X) \tag{B7}$$

$$= \mathrm{ad}_X \left(\mathrm{Ad}_{e^{tX}} Y\right). \tag{B8}$$

With the boundary condition $\mathrm{Ad}_{e^{tX}} |_{t=0} = \mathrm{Id}$ we find that the above differential equation is solved by

$$\mathrm{Ad}_{e^X} = e^{\mathrm{ad}_X} \tag{B9}$$

at $t = 1$. Consider now the following parameterized matrix function $Y : \mathbb{R} \times \mathbb{R} \to G$,

$$Y(s, t) = e^{-sX(t)}\frac{\partial}{\partial t}e^{sX(t)}, \tag{B10}$$

where $X(t)$ is a curve on $\mathfrak{g}$. We then find

$$\frac{\partial Y(s, t)}{\partial s} = e^{-sX(t)}(-X(t))\frac{\partial}{\partial t}e^{sX(t)} + e^{-sX(t)}\frac{\partial}{\partial t}(X(t)e^{sX(t)}) \tag{B11}$$

$$= -e^{-sX(t)}X(t)\frac{\partial}{\partial t}e^{sX(t)} + e^{-sX(t)}X(t)\frac{\partial}{\partial t}e^{sX(t)} + e^{-sX(t)}\frac{dX(t)}{dt}e^{sX(t)} \tag{B12}$$

$$= e^{-sX(t)}\frac{dX(t)}{dt}e^{sX(t)} \tag{B13}$$

$$= \mathrm{Ad}_{e^{-sX(t)}}\frac{dX(t)}{dt}. \tag{B14}$$

Then with equation Eq. (B9), we find

$$\frac{\partial Y(s, t)}{\partial s} = e^{-\mathrm{ad}_{sX(t)}}\frac{dX(t)}{dt}. \tag{B15}$$

Using that $Y(0, t) = 0$, we find by integration

$$Y(1, t) = \int_0^1 ds\, \frac{\partial Y(s, t)}{\partial s}. \tag{B16}$$

Estimating the above integral forms the basis of the stochastic parameter-shift rule (see App. 1). Continuing,

$$Y(1, t) = \int_0^1 ds \sum_{n=0}^{\infty} \frac{(-1)^n s^n}{n!}(\mathrm{ad}_X)^n\frac{dX(t)}{dt} \tag{B17}$$

$$= \left[\sum_{n=0}^{\infty} \frac{(-1)^n s^{n+1}}{(n+1)!}(\mathrm{ad}_X)^n\right]_{s=0}^{s=1}\frac{dX(t)}{dt} \tag{B18}$$

$$= \left(\sum_{n=0}^{\infty} \frac{(-1)^n}{(n+1)!}(\mathrm{ad}_X)^n\right)\frac{dX(t)}{dt}. \tag{B19}$$

Hence we see that

$$\frac{d}{dt}e^{X(t)} = e^{X(t)}Y(1,t) = e^{X(t)}\left(\sum_{n=0}^{\infty}\frac{(-1)^n}{(n+1)!}(\mathrm{ad}_X)^n\right)\frac{dX(t)}{dt}, \tag{B20}$$

which gives Eq. (3).

Note that at this point the Baker-Campbell-Hausdorff formula can be derived with Eq. (B20) by considering the derivative of

$$e^{Z(t)} = e^{tX}e^{tY}, \tag{B21}$$

and subsequent integration of the derivative of $Z(t)$ [77].

C Connection between Riemannian and standard gradient flow on $\mathrm{SU}(N)$

Here, we highlight the connection between a Riemannian gradient flow on $\mathrm{SU}(N)$ with respect and Eq. (11) [48, 49]. We take the Hilbert-Schmidt inner product $\langle A, B\rangle = \mathrm{Tr}\{A^\dagger B\}$ as a Riemannian metric on $\mathrm{SU}(N)$ and write the cost as

$$C(U) = \mathrm{Tr}\{U\rho U^\dagger H\}. \tag{C1}$$

The differential of the cost $dC(U) : T_U\mathrm{SU}(N) \to \mathbb{R}$, evaluated at the point $U\Omega$, where $\Omega \in \mathfrak{su}(N)$ and $U \in \mathrm{SU}(N)$, is then

$$dC(U)(U\Omega) = d(\mathrm{Tr})(U\rho U^\dagger H) \circ d(U\rho U^\dagger H)(U\Omega) \tag{C2}$$

$$= \mathrm{Tr}\{d(U)\rho U^\dagger H + U\rho d(U^\dagger)H\}(U\Omega) \tag{C3}$$

$$= \mathrm{Tr}\{U\Omega\rho U^\dagger H + U\rho\Omega^\dagger U^\dagger H\} \tag{C4}$$

$$= \mathrm{Tr}\{\rho U^\dagger HU\Omega - U^\dagger HU\rho\Omega\} \tag{C5}$$

$$= \mathrm{Tr}\{[\rho, U^\dagger HU]\Omega\} \tag{C6}$$

$$= \langle -U[\rho, U^\dagger HU], U\Omega\rangle \tag{C7}$$

via the chain rule and using that $\langle A, B\rangle = \langle UA, UB\rangle$. The compatibility condition for the Riemannian gradient tells us that

$$dC(U)(U\Omega) = \langle\mathrm{grad}\ C(U), U\Omega\rangle. \tag{C8}$$

Hence we can identify

$$\mathrm{grad}\ C(U) = -U[\rho, U^\dagger HU] \tag{C9}$$

with the corresponding gradient flow

$$\dot{U} = \mathrm{grad}\ C(U). \tag{C10}$$

Next, we follow the results of [78] in our notation. Consider the cost function

$$C(X) = \mathrm{Tr}\{e^X\rho e^{-X}H\} \tag{C11}$$

where $X \in \mathfrak{su}(N)$. Note that although the minimum of the function is unchanged, the parameterization of a unitary via the Lie algebra changes the resulting gradient flow. To see this, we consider again the differential,

$$d(C \circ \exp X)(A) = d(\mathrm{Tr})(e^X\rho e^{-X}H) \circ d(e^X\rho e^{-X}H)(A) \tag{C12}$$

$$= \mathrm{Tr}\{d(e^X)\rho U^\dagger H + U\rho d(e^{-X})H\}(A), \tag{C13}$$

where now $d(C \circ \exp X) : \mathfrak{su}(N) \to \mathbb{R}$ and $A \in \mathfrak{su}(N)$. We can now make use of the result in Eq. (B20),

$$d(e^X)(Y) = e^X \left(\sum_{n=0}^{\infty} \frac{(-1)^n}{(n+1)!} (\mathrm{ad}_X)^n \right) (Y) \tag{C14}$$

$$= e^X \Phi_X(Y) \tag{C15}$$

to obtain

$$d(C \circ \exp X)(A) = \mathrm{Tr}\left\{ e^X \Phi_X(A) \rho e^{-X} H + e^X \rho \Phi_X^\dagger(A) e^{-X} H \right\}. \tag{C16}$$

Using that $\Phi_X^\dagger(A) = -\Phi_X(A)$ and $\Phi_X^\dagger(\Phi_X(A)) = A$, we then have

$$d(C \circ \exp X)(A) = \mathrm{Tr}\left\{ \left[\rho, e^{-X} H e^X \right] \Phi_X(A) \right\} \tag{C17}$$

$$= \langle -\left[\rho, e^{-X} H e^X \right], \Phi_X(A) \rangle \tag{C18}$$

$$= \langle \Phi_X(\left[\rho, e^{-X} H e^X \right]), A \rangle. \tag{C19}$$

Hence the gradient on $\mathfrak{su}(N)$ is

$$\mathrm{grad}\, C(X) = \Phi_X(\left[\rho, e^{-X} H e^X \right]). \tag{C20}$$

We therefore see that only when $\Phi_X = I$ do we obtain the Riemannian gradient on $\mathrm{SU}(N)$; hence only if $X \in \mathfrak{g}$ where $\mathfrak{g}$ is abelian. The optimization path followed by optimizing the parameters of an $\mathbb{SU}(N)$ gate is thus different from the one following a Riemannian gradient descent on $\mathrm{SU}(N)$.

D The generalized parameter-shift rule

When using our $\mathbb{SU}(N)$ gate in an application that involves gradient-based optimization, like demonstrated in the numerical experiments in this work, we require calculating the partial derivatives in Eq. (11). Here we provide the details for how this can be achieved in practice via the generalized parameter-shift rule (GPSR) [8, 7, 9]. Without loss of generality, we rewrite the cost function in Eq. (9) as

$$C(t) = \mathrm{Tr}\left\{ \tilde{H} e^{t\Omega} \tilde{\rho} e^{-t\Omega} \right\}, \tag{D1}$$

where we absorbed the rest of the circuit into $\tilde{\rho}$, $\tilde{H}$ and fixed any other parameters in the circuit. Computing the derivative of Eq. (D1) with respect to t is equivalent to the problem of finding the gradient in Eq. (11) at $t = 0$. For the numerical experiments in this paper we make use of the particular implementation of the GPSR in [9] as well as the alternative method outlined in App. 3.

The skew-Hermitian operator Ω in Eq. (D1) has (possibly degenerate) eigenvalues $\{i\lambda_j\}$. We define the set of unique spectral gaps as $\Gamma = \{|\lambda_j - \lambda_{j'}|\}$ where $j' > j$. Note that for d distinct eigenvalues, the number of unique spectral gaps R is bounded via $R \leq d(d-1)/2$. We relabel every unique spectral gap with an integer, i.e. we write $\Delta_r \in \Gamma$, and define the corresponding vector $\boldsymbol{\Delta} = (\Delta_1, \ldots, \Delta_R)$. We pick a set of parameter shifts that are equidistant and create a vector of R shifts $\boldsymbol{\delta} = (\delta_1, \ldots, \delta_R)$ where

$$\delta_n = \frac{(2n-1)\pi}{4R}, \quad n = 1, \ldots, R. \tag{D2}$$

Next, we create the length R cost vector $\boldsymbol{c}$ and the $R \times R$ matrix M

$$\boldsymbol{c} = \begin{pmatrix} C(\delta_1) - C(-\delta_1) \\ C(\delta_2) - C(-\delta_2) \\ \vdots \\ C(\delta_R) - C(-\delta_R) \end{pmatrix}, \quad \mathrm{M}(\boldsymbol{\delta}) = \begin{pmatrix} 2\sin(\delta_1\Delta_1) & \ldots & 2\sin(\delta_1\Delta_R) \\ 2\sin(\delta_2\Delta_1) & \ldots & 2\sin(\delta_2\Delta_R) \\ \vdots & \vdots & \vdots \\ 2\sin(\delta_R\Delta_1) & \ldots & 2\sin(\delta_R\Delta_R) \end{pmatrix}. \tag{D3}$$

We then calculate the coefficient vector as

$$\boldsymbol{r} = (\mathrm{M}(\boldsymbol{\delta}))^{-1} \cdot \boldsymbol{c}, \tag{D4}$$

which finally gives the gradient

$$\left. \frac{dC(x)}{dt} \right|_{t=0} = \boldsymbol{\Delta} \cdot \boldsymbol{r}. \tag{D5}$$

Since the final gradient is exact, finite shot estimates of all $c(\delta_n)$'s will produce an unbiased estimate of $dC(x)/dt$,

$$\left. \frac{dC(x)}{dt} \right|_{t=0} = \boldsymbol{\Delta} \cdot (\mathrm{M}(\boldsymbol{\delta}))^{-1} \cdot \mathbb{E}\left[\boldsymbol{c}\right], \tag{D6}$$

where we pulled out $\boldsymbol{\Delta}$ and $(\mathrm{M}(\boldsymbol{\delta}))^{-1}$ since they are constant. The difficulty of obtaining an accurate estimate of the gradient is determined by the variance of this estimator, which is given by

$$\mathrm{Var}\left[\left. \frac{dC(x)}{dt} \right|_{t=0} \right] = (\boldsymbol{\Delta} \cdot (\mathrm{M}(\boldsymbol{\delta}))^{-1})^{\odot 2} \cdot \left(\mathbb{E}\left[\boldsymbol{c}^{\odot 2}\right] - \mathbb{E}\left[\boldsymbol{c}\right]^{\odot 2} \right), \tag{D7}$$

where we used $\odot 2$ to emphasize that the squares are taken elementwise. We assume that the estimates for each shifted circuit obey normal statistics and so since these are independent, we can write

$$\mathbb{E}\left[\boldsymbol{c}^{\odot 2}\right] - \mathbb{E}\left[\boldsymbol{c}\right]^{\odot 2} \approx \frac{1}{N_{\mathrm{shots}}} \begin{pmatrix} \sigma^2(\delta_1) + \sigma^2(-\delta_1) \\ \sigma^2(\delta_2) + \sigma^2(-\delta_2) \\ \vdots \\ \sigma^2(\delta_R) + \sigma^2(-\delta_R) \end{pmatrix}, \tag{D8}$$

where $\sigma^2(\pm\delta_n)$ is the variance of the cost for each shifted circuit. If we assume that the dependence of σ on the shifts is mild, i.e., $\sigma(\delta) \approx \sigma_0$ then the total variance will only depend on the prefactor. Setting the estimate $\mathbb{E}\left[\boldsymbol{c}^{\odot 2}\right] - \mathbb{E}\left[\boldsymbol{c}\right]^{\odot 2} = (\sigma_0^2, \sigma_0^2, \ldots, \sigma_0^2)$ then finally gives

$$\mathrm{Var}\left[\left. \frac{dC(x)}{dt} \right|_{t=0} \right] \approx 2\sigma_0^2 \left(\sum_n \Delta_n \mathrm{M}_{nm}^{-1}(\boldsymbol{\delta}) \right)^2. \tag{D9}$$

One can minimize this quantity with respect to $\boldsymbol{\delta}$ to find the optimal set of shifts for the gradient estimation [9].

E   Alternative differentiation of $\mathrm{SU}(N)$ gates

In this section we summarize a number of alternative differentiation techniques that may be applied to the presented $\mathbb{SU}(N)$ gates. In particular, we discuss the stochastic parameter-shift rule, which was created for multi-parameter gates, finite differences as a standard tool in numerical differentiation, as well as an alternative to the general parameter-shift rule above which also exploits the notion of effective generators.

1   The Stochastic parameter-shift rule

The stochastic parameter-shift rule [6] relies on the following operator identity [38]

$$\frac{\partial e^{Z(x)}}{\partial x} = \int_0^1 ds \, e^{sZ(x)} \frac{\partial Z(x)}{\partial x} e^{(1-s)Z(x)}, \tag{E1}$$

for any bounded operator $Z(x)$. We now fix all parameters θ_m for $m \neq l$ and rewrite the cost in Eq. (9) as

$$c(x) = \mathrm{Tr}\left\{ H e^{i(xG_l + A')} \rho e^{-i(xG_l + A')} \right\}, \quad A' \equiv \sum_{m \neq l} \theta_m G_m. \tag{E2}$$

Then, if we take $Z(x)$ to be the operator $Z(x) = i(xG_l + A')$ we can construct the gradient of Eq. (E2) as

$$\frac{\partial c(x)}{\partial x} = \int_0^1 ds \left(C_+(x, s) - C_-(x, s) \right), \tag{E3}$$

where

$$C_\pm(x, s) = \mathrm{Tr}\left\{ H V_\pm(x) \rho V_\pm^\dagger(x) \right\} \tag{E4}$$

$$V_\pm(x) = e^{is(xG_l + A')} e^{\pm \frac{\pi}{4} G_l} e^{i(1-s)(xG_l + A')}. \tag{E5}$$

Hence, similar to our method, the gradient evaluation requires adding gates to the circuit and evaluating the new circuit. However, the stochastic parameter-shift rule comes at a significant cost: the evaluation of the integral in Eq. (E3). In practice, one approximates this integral by sampling values of s uniformly in the interval $(0, 1)$ and then calculating the costs $C_\pm(x, s)$ with a finite-shot estimate. Although this produces an unbiased estimator, we find that the variance of this estimator is larger than ours, see Fig. 4. In addition, this method leads to a bigger number of unique circuits to compute the derivative, increasing the compilation overhead for both hardware and simulator implementations.

2   Finite differences

Finite differences are widely used to differentiate functions numerically. We briefly discuss this method in the context of variational quantum computation (VQC) and refer the reader to recent works comparing and optimizing differentiation techniques for VQC [79, 46].

In particular, we consider the central difference recipe

$$\partial_{\mathrm{FD}, \theta_j} C(\boldsymbol{\theta}) = \frac{1}{\delta} \left[C\left(\boldsymbol{\theta} + \frac{\delta}{2} \boldsymbol{e}_j \right) - C\left(\boldsymbol{\theta} - \frac{\delta}{2} \boldsymbol{e}_j \right) \right], \tag{E6}$$

where δ is a freely chosen shift parameter and $\boldsymbol{e}_j$ is the jth canonical basis vector. This recipe is an approximation of $\partial_{\theta_j} C(\boldsymbol{\theta})$, making the corresponding estimator on a shot-based quantum computer biased. This bias, which depends on δ, has to be traded off against the variance of the estimator, which grows approximately with δ^{-2}.

In classical computations, the numerical precision cutoff plays the role of the variance. Due to the high precision in classical computers, this leads to optimal shifts $\delta \ll 1$, which allows treating the bias to leading order in δ and thus enables rough estimates of the optimal δ^* in advance. On a quantum computer, however, the variance typically is more than ten orders of magnitude larger, leading to a very different δ^*, which furthermore depends on the function and derivative values. As a consequence, shifts of $\mathcal{O}(1)$ become a reasonable choice, highlighting the similarity of the central difference recipe to the two-term parameter-shift rule [79].

As a demonstration of the above, and in preparation for the numerical experiments shown in Figs. 3 and 4, we compute the central difference gradient for a random single-qubit Hamiltonian, a single $\mathbb{SU}(2)$ gate $U(\boldsymbol{\theta}) = \exp(iaX + ibY)$ and $\delta \in \{0.5, 0.75, 1.0\}$. For this, we evaluate the mean and standard error 50 times and show the difference to the exact derivative in Fig. E1. As expected, we observe that the bias increases with δ and that the variance is suppressed with larger δ. We determine $\delta = 0.75$ to be a reasonable choice for the purpose of the demonstration in Figs. 3 and 4, but stress that for any other circuit, qubit count, Hamiltonian, and even for a different parameter position $\boldsymbol{\theta}$ for this circuit, the optimal shift size needs to be determined anew.
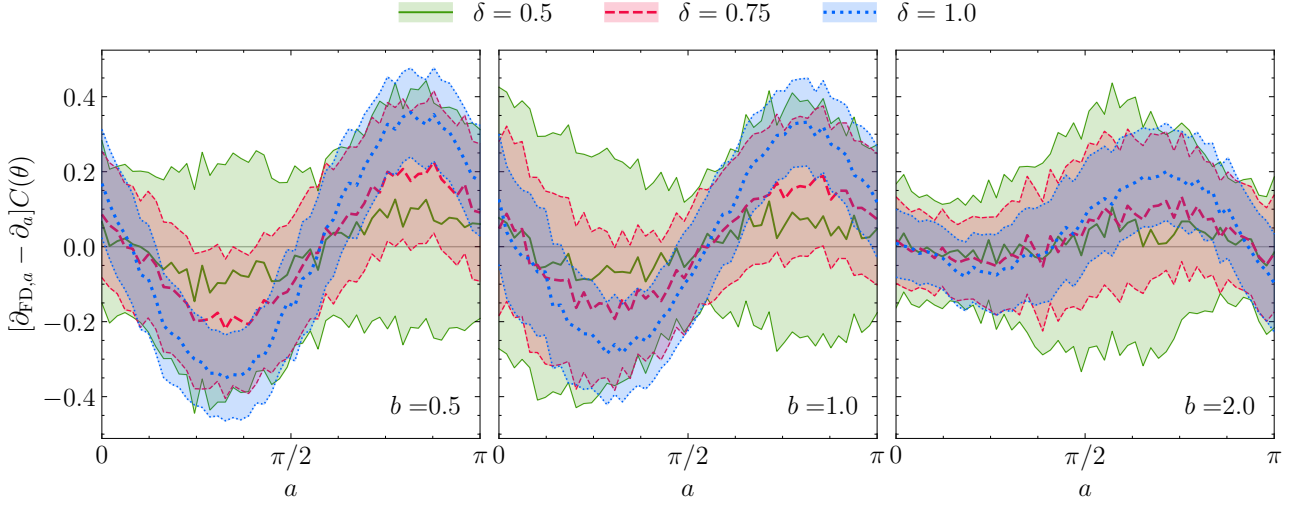
Figure E1: Error of the central difference gradients with $\delta = 0.5, 0.75, 1.0$ for the single-qubit example from Figs. 3 and 4. The value of the second parameter again is fixed to $b = 0.5, 1.0, 2.0$ in the panels (from left to right). The shift parameter δ influences the strengths of bias and variance, leading to a trade-off. For smaller δ, the variance is enhanced due to the coefficients in Eq. (E6) that scale with δ^{-1}. For larger δ, the bias based on the approximate nature of Eq. (E6) is increased. We find $\delta = 0.75$ to be a reasonable choice for this particular circuit, Hamiltonian and parameter position $\boldsymbol{\theta}$.

3 Decomposing effective generators for differentiation

In Algorithm 1 we suggest to use the generalized parameter-shift rule [7, 8, 9] in order to compute the partial derivatives $\frac{\partial}{\partial \theta_l} C(\boldsymbol{\theta})$ independently. In addition, Theorem 2 bounds the number of frequencies occurring in the univariate auxiliary cost function $C(t) = \operatorname{Tr}\left\{U(\boldsymbol{\theta})e^{t\Omega_l(\boldsymbol{\theta})}\rho e^{-t\Omega_l(\boldsymbol{\theta})}U^\dagger(\boldsymbol{\theta})H\right\}$ and the corresponding number of parameter shifts required during differentiation.

Realizing the shift rule requires us to implement not only $U(\boldsymbol{\theta})$—which is necessary to compute $C(\boldsymbol{\theta})$ itself—but also the gate $e^{t_r\Omega_l(\boldsymbol{\theta})}$ for $2R$ shift values t_r and each Ω_l separately. Alternatively, we may follow the approach to decompose all effective generators Ω_l and compute the derivative as a linear combination of the derivatives for simpler auxiliary gates, similar to [7]. In particular, we again choose the Pauli basis of $\mathfrak{su}(N)$ for this decomposition.

Decompose the effective generators $\Omega_l(\boldsymbol{\theta})$ as

$$\Omega_l(\boldsymbol{\theta}) = \sum_m \omega_{lm}(\boldsymbol{\theta})G_m, \qquad \omega_{lm}(\boldsymbol{\theta}) = \frac{1}{N}\operatorname{Tr}\{G_m\Omega_l(\boldsymbol{\theta})\}. \tag{E7}$$

Note that the coefficients are purely imaginary due to the skew-Hermiticity of $\Omega_l(\boldsymbol{\theta})$. The partial derivative we are interested in can then be written as

$$\frac{\partial}{\partial \theta_l}C(\boldsymbol{\theta}) = \operatorname{Tr}\left\{HU(\boldsymbol{\theta})\left[\sum_{m=1}^d \omega_{lm}(\boldsymbol{\theta})G_m, \rho\right]U^\dagger(\boldsymbol{\theta})\right\} \tag{E8}$$

$$= \sum_m \omega_{lm}(\boldsymbol{\theta})\operatorname{Tr}\left\{HU(\boldsymbol{\theta})[G_m, \rho]U^\dagger(\boldsymbol{\theta})\right\} \tag{E9}$$

$$= \sum_m \omega_{lm}(\boldsymbol{\theta})2i\frac{\mathrm{d}}{\mathrm{d}t}\operatorname{Tr}\left\{HU(\boldsymbol{\theta})\left[\exp\left\{-i\frac{t}{2}G_m\right\}, \rho\right]U^\dagger(\boldsymbol{\theta})\right\}\bigg|_{t=0} \tag{E10}$$

$$= \sum_m \widetilde{\omega}_{lm}(\boldsymbol{\theta})\frac{\mathrm{d}}{\mathrm{d}t}C_{G_m}(\boldsymbol{\theta}, t)\big|_{t=0}. \tag{E11}$$

Here we abbreviated $\widetilde{\omega}_{lm}(\boldsymbol{\theta}) = 2i\omega_{lm}(\boldsymbol{\theta})$ and wrote $C_{G_m}(\boldsymbol{\theta}, t)$ for the cost function with a rotation gate with parameter $-t/2$ about G_m inserted before $U(\boldsymbol{\theta})$. This modified cost function can be differentiated

with respect to t using the original two-term parameter-shift rule, as the inserted gate is generated by (the multiple of) a Pauli string.

The above linear combination of Pauli rotation derivatives can be reused for all partial derivatives, so that the full gradient for one $\mathbb{SU}(N)$ gate is given by

$$\nabla C(\boldsymbol{\theta}) = \widetilde{\omega}(\boldsymbol{\theta}) \cdot \mathbf{d}C, \tag{E12}$$

$$\mathbf{d}C = \begin{pmatrix} \frac{\mathrm{d}}{\mathrm{d}t} C_{G_1}(\boldsymbol{\theta}, t)\big|_{t=0} \\ \vdots \\ \frac{\mathrm{d}}{\mathrm{d}t} C_{G_d}(\boldsymbol{\theta}, t)\big|_{t=0} \end{pmatrix}. \tag{E13}$$

So far we did not discuss the number of Pauli strings occurring in the decomposition of the generators Ω_l. As can be seen from Eq. (6) and our definition of the DLA in Section 5, this number is bounded by the size of the DLA, and we again remark that this bound will be saturated for most values of $\boldsymbol{\theta}$. As two shifts are required for each Pauli rotation, the gradient $\nabla C(\boldsymbol{\theta})$ can thus be computed using $2 \dim \mathcal{L}(A(\boldsymbol{\theta}))$ circuits, using Pauli rotations from the DLA and, e.g., shift angles $\pm\frac{\pi}{2}$.

As we only required a linear decomposition of Ω_l, any other basis for the DLA may be used as well, potentially allowing for fewer shifted circuits or different inserted gates that may be more convenient to realize on hardware.

F Gate speed limit

The following Lemmas are used in Section 4.1.

Lemma 3. *For Hamiltonians of the form $H = \sum_m \theta_m G_m$, where G_m are strings of $\log_2 N$ Pauli operators, $\mathrm{Tr}\{H^2\} = N \sum_m \theta_m^2$.*

Proof. All Pauli strings G_m are orthonormal with respect to the trace inner product, $\mathrm{Tr}\left(G_m^\dagger G_n\right) = \delta_{n,m} N$. Using this gives

$$\mathrm{Tr}\left\{H^2\right\} = \sum_{m,n} \theta_m \theta_n \, \mathrm{Tr}\{G_m G_n\} \tag{F1}$$

$$= N \sum_{m,n} \theta_m \theta_n \delta_{n,m} \tag{F2}$$

$$= N \sum_m \theta_m^2. \tag{F3}$$

$\square$

Lemma 4. *The length of a smooth curve on the Riemannian manifold $\mathrm{SU}(N)$, with metric $g(x,y) = \mathrm{Tr}\{x^\dagger y\}$, for a time-independent Hamiltonian H after a fixed time τ only depends on the norm of H and τ.*

Proof. The unitary evolution of $U(\boldsymbol{\theta}; t)$, parameterized by t, corresponds to a smooth curve on $\mathrm{SU}(N)$ with length according to the Riemannian metric, $g(x,y) = \mathrm{Tr}\{x^\dagger y\}$ [80]. Integrating over the metric norm through the tangent spaces from $t = 0$ to final time τ gives the path length,

$$L[U(\boldsymbol{\theta}; t), \tau] = \int_{t=0}^\tau ds \tag{F4}$$

$$= \int_0^\tau \sqrt{g(\dot{U}(\boldsymbol{\theta}; t), \dot{U}(\boldsymbol{\theta}; t))} dt \tag{F5}$$

$$= \int_0^\tau \sqrt{\mathrm{Tr}\left\{\dot{U}^\dagger(\boldsymbol{\theta}; t)\dot{U}(\boldsymbol{\theta}; t)\right\}} dt, \tag{F6}$$

where $\dot{U} = \frac{dU}{dt}$. From Schrödinger evolution,

$$\frac{dU(\boldsymbol{\theta}; t)}{dt} = -iHU(\boldsymbol{\theta}; t), \tag{F7}$$

we find

$$L[U(\boldsymbol{\theta}; t), \tau] = \tau\sqrt{\text{Tr}\{H^2\}}, \tag{F8}$$

for time-independent Hamiltonians. Therefore for all Hamiltonians with fixed norm $\text{Tr}\{H^2\}$, the path distance travelled after time τ is the same regardless of the specific unitary evolution. □

Lemma 5. *The minimal path between the identity element I and a point V on the Riemannian manifold of $\text{SU}(N)$, with metric $g(x, y) = \text{Tr}\{x^\dagger y\}$, is a geodesic curve $\gamma(t) = e^{Xt}$ with $X \in \mathfrak{su}(N)$.*

Proof. From Proposition 3.10 [81], the geodesics of $\text{SU}(N)$ are the one-parameter subgroups, given by $\gamma(t) = e^{Xt}$. In general, multiple geodesics curves can give V from the identity. The minimal path is the curve with the minimum length. Since geodesic curves are the extrema of the path length functional [82, Lemma 9.3], there must exist a geodesic, which is of the form $\gamma(t) = e^{Xt}$, that is the minimal path to V.

□

1 Proof of Theorem 1

We restate Theorem 1 again for convenience.

Theorem 1. *For unitary gates generated by normalized time-independent Hamiltonians, consider a general circuit decomposition of two gates $U(\boldsymbol{\phi}^{(2)}; t_2)U(\boldsymbol{\phi}^{(1)}; t_1)$. There exists an equivalent evolution with an $\mathbb{SU}(N)$ gate $U(\boldsymbol{\theta}; t_g) = U(\boldsymbol{\phi}^{(2)}; t_2)U(\boldsymbol{\phi}^{(1)}; t_1)$, with evolution time t_g, such that*

$$t_g \le t_1 + t_2,$$

with equality if $\boldsymbol{\phi}^{(1)} + \boldsymbol{\phi}^{(1)} = \boldsymbol{\theta}$.

Proof. The product $U(\boldsymbol{\phi}^{(2)}; t_2)U(\boldsymbol{\phi}^{(1)}; t_1)$ corresponds to a specific point V on the manifold $\text{SU}(N)$. By Lemma 5, there exists a geodesic between the identity I and V, given by the curve e^{Xt} that is of minimal length. We can parameterize this geodesic as $U(\boldsymbol{\phi}^{(2)}; t_g) = \exp\{\bar{A}(\boldsymbol{\theta})t_g\}$, which is always possible since $A(\boldsymbol{\theta})$ parameterizes an arbitrary point in $\mathfrak{su}(N)$ and is an $\mathbb{SU}(N)$ gate. By Lemma 4, the length of this path only depends on the norm of $\bar{A}(\boldsymbol{\theta})$, which is 1, and on t_g, which gives

$$t_g = L[U(\boldsymbol{\theta}; t), t_g)]. \tag{F9}$$

Since this path is minimal, we have

$$t_g \le t_1 + t_2 \tag{F10}$$

with equality if $\boldsymbol{\phi}^{(1)} + \boldsymbol{\phi}^{(1)} = \boldsymbol{\theta}$. □

1.1 Special case of $\text{SU}(2)$

In the following we give the additional time for decomposing an optimal $\mathbb{SU}(2)$ gate into two gates. By *optimal*, we refer to the geodesic along the minimal path length curve – see Lemma 5.

We consider the optimal $\mathbb{SU}(2)$ gate $U(\boldsymbol{\theta}; t_g)$ with geodesic evolution time t_g together with a decomposition $U(\boldsymbol{\phi}^{(2)}; t_2)U(\boldsymbol{\phi}^{(1)}; t_1) = U(\boldsymbol{\theta}; t_g)$. The decomposed circuit is given by two unitary evolutions. Each individual evolution $U(\tilde{\boldsymbol{\phi}}^{(\nu)}; t_\nu)$ is a U(1) rotation such that only a single basis element is required. With two rotations, the overall evolution is an element of SU(2). The corresponding $\mathfrak{su}(2)$ algebra is spanned by three basis elements—the three Pauli matrices for example. The two rotations

can be represented as lying on a Bloch sphere. A unitary transformation, $K \in \mathrm{SU}(N)$, can therefore be applied to the evolution such that $U(\tilde{\boldsymbol{\phi}}^{(\nu)}; t_\nu) = KU(\boldsymbol{\phi}^{(\nu)}; t_\nu)K^\dagger$ for

$$\tilde{\boldsymbol{\phi}}^{(\nu)} = \begin{pmatrix} \sin(\alpha_\nu)\cos(\beta_\nu) \\ \sin(\alpha_\nu)\sin(\beta_\nu) \\ \cos(\alpha_\nu) \end{pmatrix}, \tag{F11}$$

where $\nu = 1, 2$, with α_ν and β_ν parameterizing the rotations. By construction $\tilde{\boldsymbol{\phi}}^{(\nu)} \cdot \tilde{\boldsymbol{\phi}}^{(\nu)} = 1$ is normalised for all parameters α_ν and β_ν. The same transformation K defines $U(\tilde{\boldsymbol{\theta}}; t_g) = KU(\boldsymbol{\theta}; t_g)K^\dagger$ and gives the same relationship $U(\tilde{\boldsymbol{\theta}}; t_g) = U(\tilde{\boldsymbol{\phi}}^{(2)}; t_2)U(\tilde{\boldsymbol{\phi}}^{(1)}; t_1)$. This is straightfoward to show

$$U(\tilde{\boldsymbol{\theta}}; t_g) = KU(\boldsymbol{\theta}; t_g)K^\dagger \tag{F12}$$

$$= KU(\boldsymbol{\phi}^{(2)}; t_2)U(\boldsymbol{\phi}^{(1)}; t_2)K^\dagger \tag{F13}$$

$$= KU(\boldsymbol{\phi}^{(2)}; t_2)K^\dagger KU(\boldsymbol{\phi}^{(1)}; t_2)K^\dagger \tag{F14}$$

$$= U(\tilde{\boldsymbol{\phi}}^{(2)}; t_2)U(\tilde{\boldsymbol{\phi}}^{(1)}; t_1). \tag{F15}$$

We have

$$\bar{A}(\tilde{\boldsymbol{\phi}}^{(\nu)}) = \sin(\alpha_\nu)\cos(\beta_\nu)G_1 + \sin(\alpha_\nu)\sin(\beta_\nu)G_2 + \cos(\alpha_\nu)G_3, \tag{F16}$$

where we choose $G_1 = iX$, $G_2 = iY$, and $G_3 = iZ$. These basis elements of $\mathfrak{su}(2)$ generate the group $\mathrm{SU}(2)$. We also define the basis vector $\boldsymbol{G} = (G_1, G_2, G_3)$. Exponentiation therefore gives the closed-form expression

$$\exp\left\{\bar{A}(\tilde{\boldsymbol{\phi}}^{(\nu)})t_\nu\right\} = \exp\left\{\left(\tilde{\boldsymbol{\phi}}^{(\nu)} \cdot \boldsymbol{G}\right)t_\nu\right\} \tag{F17}$$

$$= \cos(t_\nu)I^{\otimes N_{\mathrm{qubits}}} + \sin(t_\nu)\tilde{\boldsymbol{\phi}}^{(\nu)} \cdot \boldsymbol{G}. \tag{F18}$$

By the group composition law of $\mathrm{SU}(2)$, the product of two exponentials in $\mathrm{SU}(2)$ also gives a closed-form expression,

$$\exp\left\{\bar{A}(\tilde{\boldsymbol{\phi}}^{(2)})t_2\right\}\exp\left\{\bar{A}(\tilde{\boldsymbol{\phi}}^{(1)})t_1\right\} = \left(\cos(t_2)I^{\otimes N_{\mathrm{qubits}}} + \sin(t_2)\tilde{\boldsymbol{\phi}}^{(2)} \cdot \boldsymbol{G}\right)\left(\cos(t_1)I^{\otimes N_{\mathrm{qubits}}} + \sin(t_1)\tilde{\boldsymbol{\phi}}^{(1)} \cdot \boldsymbol{G}\right). \tag{F19}$$

Collecting terms gives

$$\exp\left\{\bar{A}(\tilde{\boldsymbol{\phi}}^{(2)})t_2\right\}\exp\left\{\bar{A}(\tilde{\boldsymbol{\phi}}^{(1)})t_1\right\} = \left(\cos(t_1)\cos(t_2) - \tilde{\boldsymbol{\phi}}^{(1)} \cdot \tilde{\boldsymbol{\phi}}^{(2)}\sin(t_1)\sin(t_2)\right)I^{\otimes N_{\mathrm{qubits}}}$$
$$+ \left(\cos(t_2)\sin(t_1)\tilde{\boldsymbol{\phi}}^{(1)} + \cos(t_1)\sin(t_2)\tilde{\boldsymbol{\phi}}^{(2)} + i\sin(t_1)\sin(t_2)\tilde{\boldsymbol{\phi}}^{(1)} \times \tilde{\boldsymbol{\phi}}^{(2)}\right) \cdot \boldsymbol{G}. \tag{F20}$$

The total evolution is

$$\exp\left\{\bar{A}(\tilde{\boldsymbol{\phi}}^{(2)})t_2\right\}\exp\left\{\bar{A}(\tilde{\boldsymbol{\phi}}^{(1)})t_1\right\} = \exp\left\{\bar{A}(\tilde{\boldsymbol{\theta}})t_g\right\} \tag{F21}$$

$$= \cos(t_g)I^{\otimes N_{\mathrm{qubits}}} + \sin(t_g)\tilde{\boldsymbol{\theta}} \cdot \boldsymbol{G}. \tag{F22}$$

By comparison of Eqs. (F20) and (F22), we find

$$\tilde{\boldsymbol{\theta}} = \frac{1}{\sin(t_g)}\left(\cos(t_2)\sin(t_1)\tilde{\boldsymbol{\phi}}^{(1)} + \cos(t_1)\sin(t_2)\tilde{\boldsymbol{\phi}}^{(2)} + i\sin(t_1)\sin(t_2)\tilde{\boldsymbol{\phi}}^{(1)} \times \tilde{\boldsymbol{\phi}}^{(2)}\right), \tag{F23}$$

and

$$\cos(t_g) = \cos(t_1)\cos(t_2) - \tilde{\boldsymbol{\phi}}^{(1)} \cdot \tilde{\boldsymbol{\phi}}^{(2)}\sin(t_1)\sin(t_2). \tag{F24}$$

The additional evolution time is $\Delta t = t_d - t_g$, with $t_d = t_1 + t_2$ the total decomposed unitary evolution time. Due to the invariance of the scalar product, $\tilde{\boldsymbol{\phi}}^{(1)} \cdot \tilde{\boldsymbol{\phi}}^{(2)} = \boldsymbol{\phi}^{(1)} \cdot \boldsymbol{\phi}^{(2)}$, the additional time $\Delta t = t_d - t_g$ required by the decomposition is then given by

$$\Delta t = t_d - \arccos\left(\cos(t_1)\cos(t_2) - \boldsymbol{\phi}^{(1)} \cdot \boldsymbol{\phi}^{(2)}\sin(t_1)\sin(t_2)\right) \geq 0.$$

G Unique spectral gaps of Dynamical Lie Algebras

1 Proof of Theorem 2

We restate Theorem 2 here for convenience.

Theorem 2. *The number of unique spectral gaps R of $\Omega_l(\boldsymbol{\theta})$ is upper bounded by the number of roots $|\Phi|$ of any maximal semi-simple DLA,*

$$R \leq |\Phi|/2. \tag{G1}$$

In the following we set $\mathfrak{g}$ to be a semisimple Lie algebra. A subspace $\mathfrak{a} \subseteq \mathfrak{g}$ is called a subalgebra if it is closed under the Lie bracket, i.e., if $[a_1, a_2] \in \mathfrak{a}$, $\forall a_1, a_2 \in \mathfrak{a}$. Since $\mathfrak{g}$ is a semisimple Lie algebra, it always contains a subalgebra called a Cartan subalgebra [29] (Chapter 7, Definition 7.10).

Definition 1. *A Cartan subalgebra $\mathfrak{h}$ of $\mathfrak{g}$ is a subalgebra that satisfies the following conditions:*

1. For all $h_1, h_2 \in \mathfrak{h}$, $[h_1, h_2] = 0$.

2. For all $x \in \mathfrak{g}$, if $[h, x] = 0$ for all $h \in \mathfrak{h}$, then $x \in \mathfrak{h}$.

The first condition tells us that $\mathfrak{h}$ is a commutative subalgebra of $\mathfrak{g}$, while the second condition says that $\mathfrak{h}$ is maximal, i.e., there is no larger commutative subalgebra. The first step in proving Theorem 2 is to make use of the following result:

Theorem 3. *[67, Chapter VI, Theorem 1]. If $\mathfrak{g}$ is a semisimple Lie algebra, we can write $\mathfrak{g}$ as a direct sum of the root spaces $\mathfrak{g}_\alpha$:*

$$\mathfrak{g} = \bigoplus_\alpha \mathfrak{g}_\alpha, \tag{G2}$$

where

$$\mathfrak{g}_{\alpha \in \mathfrak{h}^*} = \{x \in \mathfrak{g} | \mathrm{ad}_h(x) = \alpha(h)x, \forall h \in \mathfrak{h}\}, \tag{G3}$$

and $\alpha \in \mathfrak{h}^*$ are functionals on $\mathfrak{h}$. That is, a root space is a subspace of $\mathfrak{g}$ on which the action of the adjoint representation of $\mathfrak{h}$ is described by a functional (and scalar multiplication).

The above decomposition is called a root space decomposition, which is an essential tool in classifications of Lie algebras [68, 67]. Since

$$\mathfrak{g}_0 = \{x \in \mathfrak{g} | \mathrm{ad}_h(x) = 0, \forall h \in \mathfrak{h}\}, \tag{G4}$$

we find that $\mathfrak{h} = \mathfrak{g}_0$ and hence

$$\mathfrak{g} = \mathfrak{h} \oplus \bigoplus_{\alpha \neq 0} \mathfrak{g}_\alpha. \tag{G5}$$

We then immediately see that

$$\dim \mathfrak{g} = \dim \mathfrak{h} + \sum_{\alpha \neq 0} \dim \mathfrak{g}_\alpha. \tag{G6}$$

We can thus relate the dimensionality of a Lie algebra to the dimensionality of its Cartan subalgebra and its weight spaces. The second step of the proof relies on identifying the unique spectral gaps of $\Omega_l(\boldsymbol{\theta})$ with the weight spaces $\mathfrak{g}_\alpha$. To achieve this, we will construct the linear operator ad_h and apply it to the eigenbasis of $\Omega_l(\boldsymbol{\theta})$ to show that the maps α can be identified with the spectral gaps of $\Omega_l(\boldsymbol{\theta})$.

Consider an element $\Omega \in \mathfrak{g}$, where $\mathfrak{g} \subseteq \mathfrak{su}(N)$ is a non-trivial subalgebra and $\mathfrak{h}$ is a Cartan subalgebra of $\mathfrak{g}$. Since $\mathfrak{h}$ is the Lie algebra of a maximally abelian group, we can represent elements of $\mathfrak{h}$ by

diagonal matrices. Since Ω is skew-Hermitian, there exists a unitary $V \in \mathrm{SU}(N)$ that diagonalizes Ω, i.e., $V^\dagger \Omega V = h$ with $h \in \mathfrak{h}$. Here, V is the matrix with columns equal to the eigenvectors v_k of Ω with corresponding eigenvalues λ_k. We can thus always choose a basis for $\mathfrak{g}$ such that Ω is diagonal. If Ω is non-degenerate, then it must be full rank, and therefore an element of $\mathfrak{h}$. All Cartan algebras are equivalent up to conjugacy, hence we can choose the matrix h to be the diagonal matrix containing the eigenvalues of Ω to represent the Cartan subalgebra $\mathfrak{h}$. We now take E_{nm} to be the matrix with entries (n, m) equal to 1 and all other entries to 0. Define the operator

$$e_{nm} = V^\dagger E_{nm} V, \tag{G7}$$

and apply ad_h to it:

$$\mathrm{ad}_h\left(e_{nm}\right) = h e_{nm} - e_{nm} h \tag{G8}$$
$$= V^\dagger \Omega E_{nm} V - V^\dagger E_{nm} \Omega V \tag{G9}$$
$$= V^\dagger h E_{nm} V - V^\dagger E_{nm} h V \tag{G10}$$
$$= (\lambda_n - \lambda_m) e_{nm}. \tag{G11}$$

This means that ad_h has the eigenvectors e_{nm} with corresponding eigenvalues $\alpha_{nm}(h) = \lambda_n - \lambda_m$ [83], and so we have identified the eigenvalue differences with the roots of the Lie algebra. We define the set of all roots as

$$\Phi = \{\lambda_n - \lambda_m, n \neq m = 1, \ldots, N\}. \tag{G12}$$

Since the dimensionality of each weight space is one [67, Chapter VI, Theorem 2(a)], we can see that

$$\sum_{\alpha \neq 0} \dim \mathfrak{g}_\alpha = |\Phi|. \tag{G13}$$

Therefore,

$$\dim \mathfrak{g} = \dim \mathfrak{h} + |\Phi|. \tag{G14}$$

We now set $\mathfrak{g} = \mathcal{L}(A(\boldsymbol{\theta}))$. If we take the absolute value of the elements of Φ, we can identify $R = |\Phi|/2$, where the factor $1/2$ is to account for double the counting of the spectral gaps. Since Ω can be degenerate in general, we obtain the inequality $R \leq |\Phi|/2$. With this, the proof of Theorem 2 is completed.

2 Examples

Here, we give several examples of maximal DLAs and their corresponding value of $|\Phi|/2$. Analogous to the main text, we choose the Pauli representation but these results should hold for any irreducible representation of $\mathfrak{su}(N)$.

1. $\mathfrak{su}(2)$: For a 1-qubit system, there are no non-trivial subalgebras, hence we can only look at the full special unitary Lie algebra $\mathfrak{su}(2)$. Any $A(\boldsymbol{\theta})$ that consists of two Pauli operators will generate this algebra, e.g.,

$$A(\boldsymbol{\theta}) = i(\theta_1 X + \theta_2 Y) \tag{G15}$$

will give $\mathcal{L}(A(\boldsymbol{\theta})) = \mathfrak{su}(2)$. A Cartan subalgebra of $\mathfrak{su}(2)$ is given by $\mathfrak{h} = \mathrm{span}\left(\{Z\}\right)$. We therefore find that $\dim \mathfrak{g} = 3$ and $\dim \mathfrak{h} = 1$ and so $|\Phi| = 2$. Hence we have $R \leq 1$ and need $2R \leq 2$ shifts. This matches the result in [5], where the parameter-shift rule was generalized from single Pauli matrices to Hermitian operators with two unique eigenvalues.

2. TFIM: A DLA that has been studied before [62, 65] is the 1D transverse field Ising-Model (TFIM) Hamiltonian:

$$A(\boldsymbol{\theta}) = i(\theta_1 X \otimes I + \theta_2 I \otimes X + \theta_3 Z \otimes Z), \tag{G16}$$

with $\mathcal{L}(A(\boldsymbol{\theta})) = \text{span}\left(\{X \otimes I, I \otimes X, Y \otimes Y, Z \otimes Z, Z \otimes Y, Y \otimes Z\}\right)$. We can take $\mathfrak{h} = \text{span}\left(\{X \otimes I, I \otimes X\}\right)$ as a Cartan subalgebra and so $\dim \mathfrak{g} = 6$ and $\dim \mathfrak{h} = 2$, which gives $|\Phi| = 4$. Hence we need (at most) 4 shifts to obtain the gradient of an operator in the DLA of the TFIM, which corresponds to $\mathfrak{so}(4)$.

3. $\mathfrak{su}(4)$: The full Lie algebra of $\mathfrak{su}(4)$ is spanned by

$$A(\boldsymbol{\theta}) = \sum_m \theta_m G_m, \tag{G17}$$

where $G_m \in \mathcal{P}^4$ are the tensor products as defined in Eq. (1). A Cartan subalgebra of $\mathfrak{su}(4)$ is given by $\mathfrak{h} = \{Z \otimes I, I \otimes Z, Z \otimes Z\}$. This means that $\dim \mathfrak{g} = 15$ and $\dim \mathfrak{h} = 3$, which gives $|\Phi| = 12$. Hence we have $R = 6$ and need 12 shifts to obtain the gradient for a general operator in $\mathfrak{su}(4)$.

In the above examples, we have only been concerned with the dimensionality of the root system. We could go one step further and look at the structure of the root systems. It turns out that there exists only a finite set of root systems, which leads to the classification of all semisimple Lie algebras (such a program was originally carried out by Dynkin [68] and is explained in most textbooks on Lie algebras [31, 83, 29]). This allows us to make the following observation about DLAs and the $\mathbb{SU}(N)$ gates in our work: there is a finite number of families of $\mathbb{SU}(N)$ gates for each N, given by the possible DLAs. Again, we emphasize that this is independent of the representation of the algebra. We summarize the above results together with the identification of the corresponding classical group in Table 1 [31] (Chapter 3, Table 3.4).

Name	dim($\mathfrak{g}$)	dim($\mathfrak{h}$)	$|\Phi|$	Classical group
$\mathfrak{su}(2)$	3	1	2	A_1
$\mathfrak{so}(4)$	6	2	4	$A_1 \times A_1 \cong D_2$
$\mathfrak{su}(4)$	15	3	12	A_2

Table 1: Examples of DLAs and the size of the root spaces. Each root system Φ can be identified with a Lie algebra of one of the classical groups A_n, B_n, C_n, D_n. The classical group D_2 corresponds to SO(4), with the corresponding Lie algebra $\mathfrak{so}(4)$ which has dimension $N(N-1)/2$.