Supervised learning of time-independent Hamiltonians for gate design

New Journal of Physics
The open access journal at the forefront of physics

Deutsche Physikalische Gesellschaft DPG
IOP Institute of Physics
Published in partnership with: Deutsche Physikalische Gesellschaft and the Institute of Physics

CrossMark

OPEN ACCESS

RECEIVED
14 December 2019

REVISED
27 March 2020

ACCEPTED FOR PUBLICATION
17 April 2020

PUBLISHED
17 June 2020

PAPER

Supervised learning of time-independent Hamiltonians for gate design

Luca Innocenti[1], Leonardo Banchi[2,3,4,5], Alessandro Ferraro[1], Sougato Bose[3] and Mauro Paternostro[1,6]

[1] Centre for Theoretical Atomic, Molecular, and Optical Physics, School of Mathematics and Physics, Queen's University Belfast, BT7 1NN Belfast, United Kingdom
[2] QOLS, Blackett Laboratory, Imperial College London, London SW7 2AZ, United Kingdom
[3] Department of Physics and Astronomy, University College London, Gower Street, WC1E 6BT London, United Kingdom
[4] Department of Physics and Astronomy, University of Florence, via G. Sansone 1, I-50019 Sesto Fiorentino (FI), Italy
[5] INFN Sezione di Firenze, via G. Sansone 1, I-50019 Sesto Fiorentino (FI), Italy
[6] Author to whom any correspondence should be addressed.

E-mail: **m/paternostro@qub.ac.uk**

Keywords: machine learning, quantum circuits, quantum computing, supervised learning

Supplementary material for this article is available online

Abstract

We present a general framework to tackle the problem of finding time-independent dynamics generating target unitary evolutions. We show that this problem is equivalently stated as a set of conditions over the spectrum of the time-independent gate generator, thus translating the task into an inverse eigenvalue problem. We illustrate our methodology by identifying suitable time-independent generators implementing Toffoli and Fredkin gates without the need for ancillae or effective evolutions. We show how the same conditions can be used to solve the problem numerically, via supervised learning techniques. In turn, this allows us to solve problems that are not amenable, in general, to direct analytical solution, providing at the same time a high degree of flexibility over the types of gate-design problems that can be approached. As a significant example, we find generators for the Toffoli gate using only *diagonal* pairwise interactions, which are easier to implement in some experimental architectures. To showcase the flexibility of the supervised learning approach, we give an example of a non-trivial *four*-qubit gate that is implementable using only diagonal, pairwise interactions.

1. Introduction

Let us consider the synthesis of a quantum operation $\mathcal{G}$ (a *gate*) from the underlying dynamics of a quantum system. Unitarity of $\mathcal{G}$ implies the existence of a Hermitian generator $\mathcal{H}_{\mathcal{G}}$ such that $\mathcal{G} = \mathrm{e}^{\mathrm{i}\mathcal{H}_{\mathcal{G}}}$ (we assume units such that the simulation time t is dimensionless, and rescale it so that the desired gate is successfully achieved at $t = 1$). Such generator will typically contain highly non-local interactions, which can be difficult to realize in a given physical setup. However, for a choice of the platform to use for the implementation of $\mathcal{G}$, it is generally possible to single out a subset Γ of physical interactions that can be realized relatively easily and inexpensively. The question that we address here is thus: *is it possible to synthesize $\mathcal{G}$ from an $\mathcal{H}_{\mathcal{G}}$ comprising operations drawn only from an assigned Γ?*

This question is very relevant, for instance, in the context of quantum simulation, where the problem of general *reachability* of a target dynamics given a set of allowed physical interactions is pivotal [1]. However, it is also important for the realization of large-scale quantum computation [2, 3], which relies on the capability of implementing entangling gates between many qubits with high fidelity. A notable case is the quantum Toffoli gate, which is the quantum counterpart of the universal reversible classical Toffoli gate [4]. When paired with a single-qubit gate, the quantum Toffoli gate is universal for quantum computation [5].

Moreover, it is optimal for quantum error correction [6–9], and is a key component for reversible arithmetic operations such as modular exponentiation [10]. Unfortunately, the natural dynamics generating a Toffoli gate involves non-local three-qubit interactions, which are not easily implemented in experimental architectures. Other important gates, such as the Fredkin gate, share the same problem. Common strategies to overcome these limitations include a suitable use of the additional processing power offered by *larger Hilbert spaces* encompassing ancillary information carriers [11], and the embedding of quantum control techniques [12, 13]. The identification of suitable alternatives to such expensive strategies for gate synthesis and simulation would represent a significant contribution to the ongoing effort towards the translation of theoretical protocols to the production line of quantum technologies [14]. It is worth stressing that the strategies put forward in this paper differ substantially from techniques such as quantum control and gate compilation. Quantum control techniques allow for time-dependent Hamiltonians, whereas gate compilation foregoes dealing directly with Hamiltonians, looking instead for sequences of gates whose combined action results in the target operation. We, on the other hand, aim to find strategies that involve a *single, time-independent Hamiltonian*. These are sometimes referred to as *single-shot* strategies. This constraint changes the task substantially, making most techniques used in the context of quantum control not seamlessly applicable. On the other hand, gate compilation requires the underlying assumption that the elements of the set to be used in order to decompose a given unitary are all reliably implementable in the architecture under consideration. This fully bypasses any concern about the dynamics necessary to actually implement a given gate. In addition, gate compilation is combinatorial in nature. Our approach to the learning of a quantum gate, instead, makes use of a set of continuous parameters which determine the final operation only through a matrix exponential operation.

In this paper we show how the interplay between analytical results and efficient numerics—as enabled by the supervised learning paradigm—provides a powerful and flexible tool to explore a wide variety of gate design and simulation scenarios.

More specifically, we identify three analytical conditions that, when met, provide a Hamiltonian $\tilde{\mathcal{H}}$ comprising only operations drawn from Γ and such that $\mathcal{G} = \exp(\mathrm{i}\tilde{\mathcal{H}})$ for a given target $\mathcal{G}$. While these conditions do in principle allow to solve the problem in its generality, the corresponding equations soon become practically intractable. Even in this case, however, our framework provides an improved ansatz, which can be used to perform numerical optimization via supervised learning.

For the numerical optimization, we showcase an original application of supervised machine learning that allows to obtain results more efficiently than alternative methods. The context in which supervised learning is commonly explored is roughly the following: given a *training dataset* consisting of pairs $(x_i, \ell_i) \in \mathcal{X} \times \mathcal{L}$ and a parametrized function (usually referred to as the *model*) $f_{\boldsymbol{\lambda}} : \mathcal{X} \mapsto \mathcal{L}$, find $\boldsymbol{\lambda}_0$ such that $f_{\boldsymbol{\lambda}_0}(x_i) \simeq \ell_i, \forall i$. This is usually done by minimising the *empirical loss function*, that is, by solving the optimization problem $\arg\min_{\boldsymbol{\lambda}} \{\sum_i L(f_{\boldsymbol{\lambda}}(x_i), \ell_i)\}$ for some *loss function* L quantifying the distance between expected and obtained outputs (typical choices being the L_2 distance and the delta function). In other words, the goal is to approximate an unknown $g : \mathcal{X} \mapsto \mathcal{L}$ from a finite number of samples. In this sense, supervised learning is akin to function fitting. Notably, this is *not* the kind of problem naturally encountered in gate design. The design of a time-independent Hamiltonian generating a target $\mathcal{G}$, given a parametrization $\tilde{\mathcal{H}}_{\boldsymbol{\lambda}}$ for the Hamiltonian, fits into the standard scenario of solving $\operatorname{argmin}_{\boldsymbol{\lambda}}\{L(\boldsymbol{\lambda})\}$ with $L(\boldsymbol{\lambda})$ the operatorial distance between $\exp(\mathrm{i}\tilde{\mathcal{H}}_{\boldsymbol{\lambda}})$ and $\mathcal{G}$. However, as we will show here, supervised learning techniques can be fruitfully applied to this kind of problems.

Optimizing $L(\boldsymbol{\lambda})$ is a daunting task, as the evaluation of $\mathrm{e}^{\mathrm{i}\tilde{\mathcal{H}}}$ requires up to $\mathcal{O}(d^3)$ operations [15] for each iteration (d is the dimension of the Hilbert space). Nonetheless, one finds that $L(\boldsymbol{\lambda})$ can be expressed as an average over random states [16]. This suggests the use of stochastic optimization techniques, commonly employed in machine learning, where one optimizes an *estimated* empirical loss function, rather than the exact one. For gate design, the resulting estimated loss is $L(\boldsymbol{\lambda}) \simeq \sum_i D\left(\exp\left(\mathrm{i}\tilde{\mathcal{H}}_{\boldsymbol{\lambda}}\right)|\psi_i\rangle, \mathcal{G}|\psi_i\rangle\right)$ with D a state distance and $|\psi_i\rangle$ a set of random states. As in supervised learning, the latter are used to train the quantum system so that its dynamics reproduces the expected action $\mathcal{G}|\psi_i\rangle$. As typically $\tilde{\mathcal{H}}$ is sparse, the computation of $\mathrm{e}^{\mathrm{i}\tilde{\mathcal{H}}_{\boldsymbol{\lambda}}}|\psi_i\rangle$ is efficient [17], unlike the full operator exponential. Moreover, stochasticity also helps to avoid local minima. When an analytic solution is not available, our framework gives a lower-dimensional representation of the loss function, which is then optimized via stochastic gradient descent (SGD) [18, 19]. Such representation is then embedded into an automatic differentiation (AD) protocol [20–23], to efficiently compute the gradients without numerical approximations. This significantly increases the efficiency of the optimization, allowing for the exploration of previously out-of-reach scenarios, and a systematic analysis of gate design and simulation problems. While the use of SGD to tackle gate design was also present in [16], our work achieves both analytical and numerical steps forwards. In particular, no analytical framework was developed in [16], and thus exact solutions could not be found. Moreover, our use of AD represents a drastic improvement in efficiency, which enables the exploration of scenarios such as the generation of Toffoli and Fredkin gates without ancillae and with only experimentally feasible interactions, and the training of four-qubit gates. Finally, we remark that our use of supervised learning for gate design also avoids some common hurdles of standard machine learning tasks: the availability of large training sets, and overfitting. As we know the target gate, generating training data is as easy as generating random states and evolving them through a gate. Furthermore, there is no risk of overfitting: an average unit fidelity ensures that the network will work on arbitrary input states.

To demonstrate such framework, we apply it to find exact and approximated gate-design strategies. In particular, we use it to devise exact Hamiltonians generating Toffoli and Fredkin gates via pairwise interactions. The same conditions provide enhanced numerical ansatzs for the design of arbitrary N-qubit gates. We present a supervised-learning optimization technique to train qubit networks, and demonstrate algorithm-training instances of three-qubit Toffoli and Fredkin gates. We find that the training algorithm can quickly find solutions with almost-perfect fidelities, when allowing only pairwise interactions. We then present an extensive exploration of training scenarios with more restrictive sets of interactions, finding approximate generators for Toffoli and Fredkin gates with good fidelities. We go beyond the three-qubit scenario by designing a four-qubit gate using only two-qubit interactions. A significant boost in performance is here made possible by the use of AD [20–23], which allows to speed-up gradient-descent-based optimization techniques in a flexible way, at the same time avoiding numerical errors and instabilities arising from numerical differentiation techniques. We also discuss the implications of our framework for problems extending beyond the field of quantum computing, and address in its potential applications to quantum communication via perfect state-transfer approaches [24, 25]. Finally, we present a systematic exploration of the possibility of finding generators for Toffoli and Fredkin gates with experimentally viable interactions, focusing on circuit-QED platforms. Remarkably, even in such restrictive scenario, we find generators with good fidelities that use interaction strengths compatible with the capabilities of state of the art technologies [26, 27].

The remainder of this paper is organized as follows. In section 2 we illustrate the general methodology underlying our approach. Section 3 is dedicated to the application of such methods to the paradigmatic cases of quantum Toffoli and Fredkin gates. In section 4 we illustrate the supervised learning approach that we have devised in order to tackle the general cases of the problems addressed by our work (which are usually not amenable to analytic solutions). Section 5 reports our conclusions and forward look, while we delegate to a set of technical appendices the details of our analytical derivations and numerical studies.

2. General methodology

We start by computing an Hamiltonian $\mathcal{H}_\mathcal{G}$ generating the target gate $\mathcal{G} = e^{i\mathcal{H}_\mathcal{G}}$. Using the spectral decomposition of $\mathcal{G}$, we have $\mathcal{H}_\mathcal{G} = -iU\ \log(\Lambda)U^\dagger$, where $\mathcal{G} = U\Lambda U^\dagger$, log denotes the principal branch of the logarithm, and Λ is the diagonal matrix with the eigenvalues of U. Fixing a branch for the logarithm makes it single-valued, and $\mathcal{H}_\mathcal{G}$ uniquely determined from $\mathcal{G}$. To distinguish this particular Hamiltonian from the other possible generators for $\mathcal{G}$, we will refer to it as the *principal generator*. In general, the generator $\mathcal{H}_\mathcal{G}$ will contain both *physical* interactions, that can be realized easily in a given physical setup, and *unphysical* ones, i.e. dynamics that are not naturally achieved in the chosen platform. Our goal is to construct a new Hamiltonian $\tilde{\mathcal{H}}_\mathcal{G}$, comprising only physical interactions, such that $\mathcal{G} = \exp(i\mathcal{H}_\mathcal{G}) = \exp(i\tilde{\mathcal{H}}_\mathcal{G})$.

We assume that $\tilde{\mathcal{H}}_\mathcal{G}$ depends on a quantity $\boldsymbol{\lambda}$ that parametrizes the set of physical interactions to be used. For instance, in a spin system, the physical interactions can be a certain subset of the possible two-body and single-body interactions, like the set of coupling strengths and local magnetic fields. In general, $\tilde{\mathcal{H}}_\mathcal{G}(\lambda)$ may also model a system where the register is coupled to auxiliary degrees of freedom, though we will focus on the case without ancillary qubits. The following three conditions are necessary and sufficient for $\tilde{\mathcal{H}}_\mathcal{G}$ to satisfy our requirements

$$\tilde{\mathcal{H}}_\mathcal{G} \text{ contains only physical interactions,} \tag{1a}$$

$$[\tilde{\mathcal{H}}_\mathcal{G}, \mathcal{H}_\mathcal{G}] = 0, \tag{1b}$$

$$Eig(\tilde{\mathcal{H}}_\mathcal{G} - \mathcal{H}_\mathcal{G}) = \{2\pi n_i\} \quad (n_i \in \mathbb{Z}). \tag{1c}$$

Equations (1b) and (1c) ensure that $\mathcal{G} = \exp(\mathrm{i}\tilde{\mathcal{H}}_{\mathcal{G}} - \mathrm{i}\mathcal{H}_{\mathcal{G}})\exp(\mathrm{i}\mathcal{H}_{\mathcal{G}})$ and $\exp(\mathrm{i}\tilde{\mathcal{H}}_{\mathcal{G}} - \mathrm{i}\mathcal{H}_{\mathcal{G}}) = \mathbb{1}$, while (1a) reiterates the constraints on $\tilde{\mathcal{H}}_{\mathcal{G}}$. While equation (1b) may seem too restrictive, this turns out to not be the case. To see this, consider the decomposition $\mathcal{G} = \sum_k \lambda_k \sum_j P_{kj}$ of a general gate $\mathcal{G}$. Here, P_{kj} is the jth trace-1 projector in the kth degenerate subsector of the eigenspace[7]. It follows that $\mathcal{H}_{\mathcal{G}}$ and $\tilde{\mathcal{H}}_{\mathcal{G}}$ can be written as $\mathcal{H}_{\mathcal{G}} = -\mathrm{i}\sum_k \log(\lambda_k) I_k$ and $\tilde{\mathcal{H}}_{\mathcal{G}} = \mathcal{H}_{\mathcal{G}} + 2\pi\sum_{k,j}\nu_{kj}P_{kj}$. Here $I_k = \sum_j P_{kj}$, which is not (in general) an identity (or diagonal) matrix, and we have used the expression $\log\lambda = \log\lambda + 2\pi i\nu$ $(\nu \in \mathbb{Z})$, applied to every term of the spectral decomposition of $\mathcal{G}$. For any $\tilde{\mathcal{H}}_{\mathcal{G}}$ we thus have $[\tilde{\mathcal{H}}_{\mathcal{G}}, \mathcal{H}_{\mathcal{G}}] = 0$.

Equations (1a–1c) considerably simplify the problem of gate synthesis, and can be used in several ways. First, they can be analytically solved in at least some situations of physical interest. Second, they can be used to produce an efficient starting point for numerical optimization techniques. The general procedure is to start from a parametrized expression for $\tilde{\mathcal{H}}_{\mathcal{G}}(\boldsymbol{\lambda})$ satisfying the condition stated in equation (1a), and then use equation (1b) to both reduce the set of possible interactions and impose constraints on the parameters. The problem then reduces to the enforcing of the constraints set by equation (1c). This is the non-trivial step in the procedure, which we will however show to be analytically solvable in at least some cases. This strategy thus reduces the task of constrained gate design into an inverse eigenvalue problem, a topic well studied in the field of numerical analysis [28]. More generally, we develop a numerical supervised learning technique to avoid having to solve directly the non-trivial eigenvalue problem posed by equation (1c). It is worth noting that, while $\tilde{\mathcal{H}}_{\mathcal{G}}$ produces the same unitary evolution given by $\mathcal{H}_{\mathcal{G}}$ at $t = 1$, the dynamics will in general be different at $0 < t < 1$.

3. Applications: Toffoli and Fredkin gates

3.1. Quantum Toffoli gate

The quantum Toffoli gate $\mathcal{U}_{\mathrm{Toff}}$ is a control-control-NOT that flips the state of the target qubit (the third qubit in our case) when the state of the two controls (first and second qubits) is $|1\rangle_1 \otimes |1\rangle_2$, and acts trivially on the target qubit otherwise. Its realization is an important step towards the construction of quantum computers [9, 29–31]. A time-independent two-body Hamiltonian that simulates $\mathcal{U}_{\mathrm{Toff}}$ with four qubits has been obtained in [16] using numerical optimization, while three qubits have only been found to make approximate and classical Toffoli gates [32]. Here, following the construction in equation (1), we find an analytic solution that requires only three qubits.

The principal generator of $\mathcal{U}_{\mathrm{Toff}}$, obtained by taking the principal value of its logarithm, is

$$\mathcal{H}_{\mathrm{Toff}} = (\pi/8)(1 - \sigma_1^z)(1 - \sigma_2^z)(1 - \sigma_3^x), \tag{2}$$

whose only three-qubit term is $\sigma_1^z\sigma_2^z\sigma_3^x$, where σ_i^α is the α Pauli matrix acting on the ith qubit. We now parametrize $\tilde{\mathcal{H}}_{\mathrm{Toff}}$ as

$$\tilde{\mathcal{H}}_{\mathrm{Toff}} = h_0\mathbb{1} + \sum_i h_i^\alpha\sigma_i^\alpha + \sum_{i,j} J_{i,j}^{\alpha,\beta}\sigma_i^\alpha\sigma_j^\beta, \tag{3}$$

where $h_0, h_i^\alpha, J_{i,j}^{\alpha,\beta}$ are real parameters. The above expression, containing 37 parameters, automatically satisfies equation (1a) in that it corresponds to an $\tilde{\mathcal{H}}_{\mathrm{Toff}}$ without three-qubit interactions. Imposing the condition in equation (1b) removes 12 parameters, leaving us with 25 of them. These are still too many to solve the inverse eigenvalue problem of equation (1c). We thus impose physically plausible assumptions on the coefficients in order to obtain a generator with a small enough number of parameters, so that equation (1c) is satisfied *and* the resulting equations solved. In particular, we set

$$\begin{aligned} &J_{12}^{xz} = J_{12}^{zx} = J_{13}^{xx} = J_{23}^{xx} = 0, \\ &J_{13}^{zx} = J_{23}^{zx} = \pi/8, \quad J_{23}^{zz} = -J_{13}^{zz}, \quad h_1^z = h_2^z = -\pi/8. \end{aligned} \tag{4}$$

The rationale behind these assumptions is to look for a generator that is diagonal with respect to the first two qubits, does not use σ_i^y operators, and does not introduce new off-diagonal interactions, on top of the ones already in the principal generator. This last assumption is useful because it implies a reduced number of parameters in $\mathcal{H}'_{\mathrm{Toff}} \equiv \tilde{\mathcal{H}}_{\mathrm{Toff}} - \mathcal{H}_{\mathrm{Toff}}$, which is the operator on which we have to impose equation (1c).

[7] The projectors are such that $\sum_j \mathrm{P}_{\mathrm{kj}} = \mathbb{1}_{\mathrm{k}}$, $P_{kj}P_{kj'} = 0$ if $j \neq j'$, and $P_{kj}P_{k'j'} = 0$ if $k \neq k'$ for any j, j'.

Note that different assumptions, leading to different solutions, are possible. Imposing the above constraints we obtain

$$\mathcal{H}'_{\text{Toff}} = (\pi/8)\sigma_1^z\sigma_2^z\sigma_3^x + (h_0 - \pi/8)\mathbb{1} + (h_3^x + \pi/8)\sigma_3^x + (J_{12}^{zz} - \pi/8)\sigma_1^z\sigma_2^z + J_{13}^{zz}(\sigma_1^z - \sigma_2^z)\sigma_3^z. \tag{5}$$

The problem is now to find values for the coefficients in equation (5) such that $\exp(\mathrm{i}\mathcal{H}'_{\text{Toff}}) = \mathbb{1}$, which is equivalent to finding coefficients such that all the eigenvalues of $\mathcal{H}'_{\text{Toff}}$ are integer multiples of 2π. Solving for $h_0, h_3^x, J_{12}^{zz}, J_{13}^{zz}$ gives several solutions. For example, we find that

$$\tilde{\mathcal{H}}_{\text{Toff}}(\nu) = \frac{\pi}{8}\left[1 + 4|\nu| - 2\sigma_3^x - \sigma_1^z - \sigma_2^z + (\sigma_1^z + \sigma_2^z)\sigma_3^x + (1 - 4|\nu|)\sigma_1^z\sigma_2^z + \sqrt{16\nu^2 - 1}(\sigma_2^z - \sigma_1^z)\sigma_3^z\right] \tag{6}$$

is a class of generators for $\mathcal{U}_{\text{Toff}}$, satisfying $\exp(\mathrm{i}\tilde{\mathcal{H}}_{\text{Toff}}(\nu)) = \mathcal{U}_{\text{Toff}}$ for all non-zero integers ν. In appendix A3 we present a larger class of generators for the Toffoli, as well as provide an in-depth analysis of how these generators produce the Toffoli gate. Moreover, in appendix A4, we discuss a possible line of reasoning to arrive to one of these generators via direct algebraic manipulation. While with this ad hoc reasoning we cannot recover the full set of generators obtainable via our framework, we still gather insight on the way non-trivial interaction coefficients emerge. Moreover, it showcases the type of sophisticated analysis needed to solve these problems without using our framework. Thus, a highly non-trivial gate such as Toffoli's, which in principle requires three-body interactions as in equation (2), can be obtained exactly *without* three-qubit interactions. Notably, although the generators for the Toffoli gate found here contain non-diagonal σ^z–σ^x interactions, which may be hard to implement, we will show in section 4 that this can be overcome using a supervised learning approach.

3.2. Quantum Fredkin gate

On a similar note, it is possible to use the framework provided by equations (1a–1c) to find a Hamiltonian that does not contain three-qubit interaction terms, and generates the Fredkin gate at suitable times. The quantum Fredkin gate $\mathcal{U}_{\text{Fred}}$ is a three-qubit gate which swaps two qubits conditionally to the first qubit being in the $|1\rangle$ state, and is of use for a number of quantum information protocols [33, 34]. A time-independent two-body Hamiltonian simulating $\mathcal{U}_{\text{Fred}}$ with four qubits has been found in [16] using numerical optimization. We find an analytic solution that requires as few as three qubits. Explicitly

$$\mathcal{H}_{\text{Fred}} = \frac{\pi}{8}\left(\mathbb{1} - \sigma_1^z\right)\left[\mathbb{1} - \sum_\alpha \sigma_2^\alpha\sigma_3^\alpha\right], \tag{7}$$

where qubit 1 is the control. We now write down the general parametrization $\tilde{\mathcal{H}}_{\mathcal{G}}(\boldsymbol{\lambda})$ for a three-qubit Hamiltonian containing only pairwise *diagonal* interactions, and imposing equation (1b) we cut the number of parameters $\boldsymbol{\lambda}$ down to 22. Imposing additional physically plausible conditions, like the symmetry of second and third qubit, we manage to reduce the number of parameters enough to solve the eigenvalue problem, finding

$$\tilde{\mathcal{H}}_{\text{Fred}} = \frac{\pi}{8}\left(\sqrt{\frac{143}{5}}\mathbb{1} + 5\sqrt{3}\sigma_1^x\right)(\sigma_2^x + \sigma_3^x) - \frac{3\pi}{8}\sum_{\alpha=x,y,z}\sigma_2^\alpha\sigma_3^\alpha + \frac{3\pi}{4}\sqrt{\frac{7}{5}}\sigma_1^z(\sigma_2^z + \sigma_3^z) + \frac{\pi}{2}\sigma_1^z + \frac{3\pi}{8}\mathbb{1}. \tag{8}$$

Therefore, also a non-trivial gate of crucial relevance such as the Fredkin gate can be implemented without time-dependent dynamics using only at most two-qubit interactions. The physical reason behind this simplification can be understood from the study of the spectral properties. For example, the gate $\mathcal{U}_{\text{Fred}}$ has only two eigenvalues, $\lambda_\pm = \pm 1$ with λ_+ having a sevenfold degeneracy. Such degeneracy makes the propagator generated by $\mathcal{H}_{\text{Fred}}$ operate in a two-level subspace. On the other hand, the spectrum of $\tilde{\mathcal{H}}_{\text{Fred}} - \mathcal{H}_{\text{Fred}}$ is $\{-4\pi, -2\pi, 0, 0, 0, 2\pi, 2\pi, 4\pi\}$, showing that the degeneracy in the spectrum of $\mathcal{H}_{\text{Fred}}$ is partially lifted when considering $\tilde{\mathcal{H}}_{\text{Fred}} - \mathcal{H}_{\text{Fred}}$, and the dynamics thus occurs in an larger effective Hilbert space. Although $\exp(\mathrm{i}t\tilde{\mathcal{H}}_{\text{Fred}})$ is non-symmetric for most of the evolution times, all the symmetries are restored at $t = 1$ and any subsequent integer times. This shows that breaking the symmetries of $\mathcal{G}$ and exploiting its degenerate space can help the gate design when restricting the set of viable interactions.

Algorithm 1. Stochastic gate design.

```
1: Choose values for the set $\boldsymbol{\lambda}_0$ for $\tilde{\mathcal{H}}_{\mathcal{G}}$, such that equation (1a) is satisfied.
2: Use equation (1b) to find a reduced set $\boldsymbol{\lambda}$ (linearly related to $\boldsymbol{\lambda}_0$).
3: repeat                                   ▷ Iteratively solve equation (1c)
4:     Generate a random set of $N_b$ input states $\{|\psi_k\rangle\}$ with $N_b$ the size of the mini-batches
       chosen beforehand.
5:     For each $k$, compute $\nabla_{\boldsymbol{\lambda}}\mathcal{F}(\boldsymbol{\lambda}, \psi_k)$. Machine learning frameworks like Theano [38],
       TensorFlow [39], or PyTorch [40] (among others) enables the calculation of gradients automatically
       from the chain rule. This avoids numerical errors arising from numerical differentiation.
6:     Update the coupling strengths $\boldsymbol{\lambda}$. We do this using the so-called momentum gradient
       descent method [41], corresponding to the following updating rule

           $\boldsymbol{v} \to \gamma\boldsymbol{v} + \eta\nabla_{\boldsymbol{\lambda}}\mathcal{F}(\boldsymbol{\lambda}, \psi_k), \quad \boldsymbol{\lambda} \to \boldsymbol{\lambda} + \boldsymbol{v}.$   (9)

       Here the learning rate $\eta$ and the momentum term $\gamma$ are hyperparameters to be chosen
       beforehand. The value of $\eta$ should decrease with the iteration number.
7: until a satisfactory value of the fidelity is obtained.
```

4. Supervised learning approach

We now describe a different methodology to solve the difficult part of equation (1). While the direct algebraic approach fails as soon as we consider more than a few parameters, and for example already fails to find solutions for the Toffoli gate with only diagonal interactions, the method we present here scales better with the number of interactions and is easily generalized to any kind of structure of the qubit network. The idea is to adopt a supervised learning approach to solve the optimization problem of finding the set of Hamiltonian parameters generating a target evolution.

The problem we address can be stated as follows. Given a target gate $\mathcal{G}$ and a parametrized Hamiltonian $\mathcal{H}(\boldsymbol{\lambda}) = \sum_i \lambda_i O_i$, where $\boldsymbol{\lambda} = \{\lambda_i\}$ is a set of real parameters and O_i are Hermitian operators, we look for the set $\boldsymbol{\lambda}_0$ such that $\exp(\mathrm{i}\mathcal{H}(\boldsymbol{\lambda}_0)) = \mathcal{G}$. This can be reframed as an optimization problem by considering the fidelity function $\mathcal{F}(\boldsymbol{\lambda}, \psi) \equiv \langle\psi|\mathcal{G}^\dagger \exp(\mathrm{i}\mathcal{H}(\boldsymbol{\lambda}))|\psi\rangle$, for an arbitrary state $|\psi\rangle$. Clearly, $\mathcal{F}(\boldsymbol{\lambda}_0, \psi) = 1$ for all $|\psi\rangle$ *iff* $\exp(\mathrm{i}\mathcal{H}(\boldsymbol{\lambda}_0)) = \mathcal{G}$. A possible approach to find such $\boldsymbol{\lambda}_0$ is to consider the average fidelity function $\bar{\mathcal{F}}(\boldsymbol{\lambda})$, defined as the average of $\mathcal{F}(\boldsymbol{\lambda}, \psi)$ over all ψ[8]. Explicit formulas for $\bar{\mathcal{F}}$ are known [35, 36], so that standard optimization methods can be used. This method is however inefficient for the problem at hand, due to the size of the parameter space. We thus turn to a different technique, exploiting how the fidelity landscape changes when changing $|\psi\rangle$ [16]. We use the fact that the only values of $\boldsymbol{\lambda}$ for which the fidelity is 1 regardless of $|\psi\rangle$ are those corresponding to our solution. Using the SGD algorithm [18, 19, 37], we implement the procedure in algorithm 1. A more detailed presentation of this algorithm is given in appendix B.To find the interaction parameters implementing a Toffoli gate, using only one-qubit evolutions and two-qubit diagonal interactions (i.e. interactions of the form $\sigma_1^\alpha\sigma_2^\alpha$), we start the numerical training from the Hamiltonian obtained by imposing equation (1b) on the parametrized Hamiltonian containing the required interactions, which has the form

$$\tilde{\mathcal{H}}_{\mathrm{Toff}} = \sum_{j=1,2} J_{j3}^{zz}(\mathbb{1} + \sigma_j^z)\sigma_3^z + J_{12}^{yy}(\sigma_1^x\sigma_2^x + \sigma_1^y\sigma_2^y) + J_{12}^{zz}\sigma_1^z\sigma_2^z$$
$$+ h_1^z\sigma_1^z + h_2^z\sigma_2^z + h_3^x\sigma_3^x + (J_{13}^{xx}\sigma_1^x + J_{23}^{xx}\sigma_2^x)(\mathbb{1} + \sigma_3^x). \tag{10}$$

Starting from equation (10), many solutions can be found, depending on the chosen initial conditions and the random states that are used at each run. In figure 1(a), several solutions are shown, proving that it is possible to implement a Toffoli gate using only pairwise diagonal interactions. This analysis can be extended to a Fredkin gate, whose generator with only diagonal pairwise interactions and commuting with the principal generator of the Fredkin is found to have at most two-body terms. Using this model as starting point for the training we again obtain several solutions, some of which are shown in figure 1(b). We then test our supervised learning framework by exploring numerically the possibility of implementing Toffoli and Fredkin gates using a more restrictive set of interactions. The results provide a series of physically-feasible interaction parameters that realize Toffoli and Fredkin gates with good fidelities, as further detailed in appendix B4.

[8] Note that in principle a more reliable figure of merit to evaluate the performance of a gate would be the diamond distance. However, in all the instances we considered, the two quantities are actually equivalent since we always obtain unit fidelity (within numerical errors).

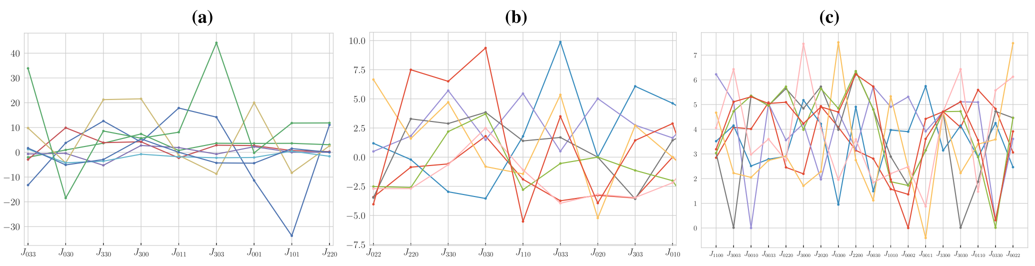

Figure 1. Eight different sets of interaction parameters generating the Toffoli gate [panel (a)] and the Fredkin one [panel (b)]. For each shown solution the training was started from the ansatz provided by equation (10), and the analogous equation for the Fredkin, respectively. Panel (c): eight sets of interaction parameters for the 'double Fredkin' gate. Each one of the shown solutions corresponds to unit fidelity up to numerical precision (that is, fidelities greater than 1×10^{-16}). We refer to appendix B4 for the details of the optimizations. To use a more uniform the notation, we denote with J_{ijk} the coefficient of the interaction parameter $\sigma_1^i\sigma_2^j\sigma_3^k$, and similarly for interactions between more than three qubits. For example, $J_{033} \equiv J_{23}^{zz}$, $J_{200} \equiv h_1^y$, and $J_{1010} \equiv J_{13}^{xx}$.

There is no need to stick to the training of three-qubit networks. To illustrate this, we provide another example of successful application of our framework, this time to implement a non-trivial unitary evolution over *four* qubits. In particular, we train a four-qubit network to implement the *doubly-controlled* Fredkin gate $\mathcal{U}_{\rm FF} \equiv |0\rangle\langle 0| \otimes \mathcal{U}_{\rm Fred} + |1\rangle\langle 1| \otimes \mathcal{U}_{\overline{\rm Fred}}$, where $\mathcal{U}_{\overline{\rm Fred}}$ denotes a Fredkin gate where the control qubit is the third one, and the first two are the targets. Such a gate can be implemented using no more than two-qubit interactions, and such set can be further restricted to only *diagonal* ones. Some instances are shown in figure 1(c). Note that these results require no ad hoc approach or ansatz, differently from the approach used to derive equations (6) and (8). In particular, this simplifies the test of questions as 'can gate *X* be implemented using only a set of interaction *Y*' without resorting to extra ad hoc calculations.

5. Conclusions

We have presented a general framework to approach constrained gate-synthesis problems. We have showed that the procedure is amenable to direct analytical solution, providing time-independent Hamiltonians generating Toffoli and Fredkin gates using only undemanding diagonal interactions and no ancillary qubits. To our knowledge, no previous attempt at such a decomposition has been reported so far. Generality can be added to our approach by powerful techniques of supervised learning of the interaction parameters, which allowed to find Hamiltonians with specified sets of interactions producing target unitary evolutions. Our approach and results are potentially of great interest to optimize experimental implementations of quantum algorithms in architectures such as linear optics and superconducting qubits. Moreover, it might open new avenues for the exploration of related problems such as Hamiltonian learning [42–44], which might be seen as addressing a question complementary to the one tackled here. In particular, it will be interesting to address whether it is possible to design a supervised-learning approach to the data available for a class of Hamiltonians that should be learnt, in line with the strategy put forward with our method, and thus devise a strategy that is competitive with respect to recently proposed Bayesian approaches [45].

Acknowledgments

This work was supported by the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013)/ERC Grant agreement No. 308253 PACOMANEDIA, the UK EPSRC (Grants No. EP/N031105/1 and EP/S000267/1), the DfE-SFI Investigator Programme (Grant 15/IA/2864), the H2020 Collaborative Project TEQ (Grant 766900), the Leverhulme Trust through the Research Project Grant UltraQuTe (Grant nr. RGP-2018-266), the Royal Society Wolfson Fellowship scheme through project ExTraQCT (RSWF\R3\183013), and the Italian MIUR 'Rita Levi Montalcini' program for your young researchers. LI is supported by Fondazione Angelo della Riccia. The data in support of the findings of this study, as well as all source code used to produce it, are available in the online repositories github.com/lucainnocenti/quantum-gate-learning-1803.07119 and github.com/lucainnocenti/quantum-gate-learning-1803.07119-Mathematica-code.

Appendix A. Analytical approach

In this section we provide a detailed description of how to apply the framework introduced in section 2 to approach gate design problems analytically.

In appendix A1 we show how our framework can be applied to tackle generic perfect state transfer problems. In appendix A2 we prove that a CNOT gate *cannot* be implemented using only *single-qubit* interactions. While this is an obvious result, the calculation can be interesting to show, in a simple case, how the eigenvalue conditions given in equation (1c) can be used to *rule out* that a gate can be implemented with a specific set of interactions. In section 3 we showed how to use our framework to derive Hamiltonians implementing the Toffoli gate using only one- and two-qubit interactions. In appendix A4 we will show how one of the solutions obtained in appendix A3 could have been obtained via direct analytical reasoning, without prior knowledge of our solution framework. This provides some insight into how the solutions actually generate the Toffoli gate, and illustrates the kind of ad hoc non-trivial reasoning that, without the use of equation (1), would have been necessary to find such solutions.

For notational convenience, we will in this section denote Pauli matrices with X_i, Y_i, Z_i, instead of σ_i^x, σ_i^y, and σ_i^z as in the main text.

A1. Perfect state transfer

A one-dimensional quantum walk is described by the Hamiltonian $\mathcal{H}_{\mathrm{W}} = \sum_{k=1}^{N-1} J_k|k\rangle\langle k-1| + B_k|k\rangle\langle k|$, where N is the length of the lattice upon which the walk takes place, J_k the transition rates between adjacent sites, B_k the local energies and $|k\rangle$ defines the state where the 'walker' is at the kth site. A quantum walk Hamiltonian $\mathcal{H}_{\mathrm{W}}$ admits perfect state transfer (PST) at time t, i.e. the initial state of the walker initially at site k is perfectly retrieved at site $N-j+1$ if $\mathrm{e}^{-\mathrm{i}t\mathcal{H}_{\mathrm{W}}} = \Xi$, where $\Xi_{kj} = \delta_{k,N-j+1}$ is the reflection matrix. Necessary and sufficient conditions for PST are well understood [24, 25]: firstly, $\mathcal{H}_{\mathrm{W}}$ has to be 'mirror-symmetric', that is $[\mathcal{H}_{\mathrm{W}}, \Xi] = 0$, and secondly the eigenvalues $\{E_k\}$ of $\mathcal{H}_{\mathrm{W}}$ should to satisfy the condition $\mathrm{e}^{\mathrm{i}E_k t} = (\pm 1)^k$.

We show now that finding the parameters $\{J_k, B_k\}$ for PST is a particular case of the Hamiltonian design problem for gate simulation. Specifically, the conditions for PST can be obtained from the construction in equation (1). In a 1D quantum walk, the 'physical' couplings are the nearest neighbour interactions, but $\mathcal{H}_\Xi = i\log(\Xi) = \pi(\Xi - \mathbb{1})/2$ is long range. Using $\mathcal{H}_{\mathrm{W}}$ as $\mathcal{H}^{\mathrm{phys}}(\lambda)$, where $\lambda = \{J_k, B_k\}$, and defining $\mathcal{H}' = \mathcal{H}_{\mathrm{W}} - \mathcal{H}_\Xi$ following condition equation (1c), where the physical interactions in $\mathcal{H}_\Xi$ have been reabsorbed into the definition of $\mathcal{H}_{\mathrm{W}}$, one finds that: (i) condition (1b) is equivalent to the mirror-symmetry request $[\mathcal{H}_{\mathrm{W}}, \Xi] = 0$; (ii) from condition (1c) and the definitions of $\mathcal{H}_{\mathrm{W}}$ and $\mathcal{H}_\Xi$, one finds that the spectrum of $\mathcal{H}_{\mathrm{W}}$ satisfies $\mathrm{e}^{\mathrm{i}E_k t} = (\pm 1)^k$—indeed the eigenvalues of Ξ are $\{0, \pi\}$, so $E_k = \pi(2n_k + 1)$ for integer n_k. It is straightforward to extend the above proof for more general transfers, such as with long-range interactions [46], or for perfect fractional revivals [47, 48].

A2. Proof that CNOT needs two-qubit terms

We here show how to use our framework to prove that a CNOT gate cannot be implemented using only one-qubit interactions. While this result is trivial, it is nonetheless interesting to show how the framework can be used to obtain this kind of impossibility results.

The spectral decomposition of the CNOT reads

$$\mathrm{CNOT} = Z_1^+ + Z_1^- X_2^+ - Z_1^- X_2^-, \tag{S1}$$

where we made a canonical choice for the basis of the three-fold degenerate eigenspace corresponding to the eigenvalue $+1$, and defined $Z_i^\pm \equiv (1 \pm Z_i)/2$, and similarly for $X_i^\pm$. More explicitly, we are considering the basis set $\{Z_1^+ Z_2^+, Z_1^+ Z_2^-, Z_1^- X_2^+\}$ made out of trace-1 projectors for the degenerate space, whereas the fourth projector is bound to be $Z_1^- X_2^-$. The corresponding principal Hamiltonian $\mathcal{H}_{\mathrm{CNOT}}$, obtained by directly taking the logarithm of equation (S1), is

$$\mathcal{H}_{\mathrm{CNOT}} = \pi Z_1^- X_2^-. \tag{S2}$$

Let us now consider what happens when the multivaluedness of the logarithm is taken into account, but no rotation of the degenerate eigenspace is performed. Considering only the factors with two-qubit interactions, the following expression is found:

$$\mathcal{H}_{\rm CNOT}/2\pi \sim \nu_{12} Z_1 Z_2 + 2\pi(1/2+\nu_{43}) Z_1 X_2, \tag{S3}$$

where here $\nu_{ij} \equiv \nu_i - \nu_j$, and $\nu_i \in \mathbb{Z}$ is the integer produced by application of the logarithm to the ith projector. Note how the $Z_1 X_2$ factor cannot be removed by *any* choice of ν_i, which could be interpreted as a proof that two-qubit interaction terms are indeed necessary to implement the CNOT gate. This, however, does not in principle preclude the possibility that an appropriate rotation of the degenerate space allows to obtain a generator with only local terms. To verify that this is not the case, we would have to consider a generic rotation R of the degenerate space, that is, an operator of the form $R = \sum_{i,j=1}^3 r_{ij} |+1_i\rangle\langle +1_j|$, with $|+1_i\rangle$ the ith eigenvector in a fixed base of the degenerate space. The problem is thus to find a unitary R and integers ν_i such that

$$\nu_1 R(Z_1^+ Z_2^+)R^\dagger + \nu_2 R(Z_1^+ Z_2^-)R^\dagger + \nu_3 R(Z_1^- X_2^+)R^\dagger + (\nu_4 + 1/2) Z_1^- X_2^-$$

does not contain two-qubit interactions. The solution of this problem is non-trivial, mostly due to the many (nine in this case) parameters characterising a general unitary R. To avoid searching solutions for such a system, we try a different approach to the problem. Let us denote with $\tilde{\mathcal{H}}$ a generator with the required properties: one that generates the same unitary as $\mathcal{H}_{\rm CNOT}$ and contains only one-qubit interaction terms. Its general form will be

$$\tilde{\mathcal{H}} = h_0 + \sum_{\alpha=1}^{3} (h_1^\alpha \sigma_1^\alpha + h_2^\alpha \sigma_2^\alpha). \tag{S4}$$

As discussed in section 2, for $\tilde{\mathcal{H}}$ to correctly generate the CNOT gate, it must commute with the principal generator $\mathcal{H}_{\rm CNOT}$. Imposing this commutativity removes most of the parameters h_i^α, leaving us with the following simplified expression:

$$\tilde{\mathcal{H}} = h_0 + h_{1,3} Z_1 + h_{2,1} X_2. \tag{S5}$$

The only missing step is now to impose the eigenvalues of $\mathcal{H}_{\rm CNOT} - \tilde{\mathcal{H}}$ to be integer multiples of 2π. This gives the following system of equations:

$$\begin{aligned}
2\pi\nu_1 &= (-\pi + 4h_0 - 4h_{1,3} - 4h_{2,1})/4, \\
2\pi\nu_2 &= (+\pi + 4h_0 + 4h_{1,3} - 4h_{2,1})/4, \\
2\pi\nu_3 &= (+\pi + 4h_0 - 4h_{1,3} + 4h_{2,1})/4, \\
2\pi\nu_4 &= (-\pi + 4h_0 + 4h_{1,3} + 4h_{2,1})/4,
\end{aligned} \tag{S6}$$

with $\nu_i \in \mathbb{Z}$. The above system can be seen to have no solution for $h_0, h_{1,3}, h_{2,1}$, therefore definitively proving that there is no rotation of the degenerate space, and integer parameters ν_i, that allow to generate the CNOT gate using only one-qubit interactions.

A3. Toffoli gate: derivation through conditions

We show here the details of how, using equation (1), we can obtain a family of Hamiltonian generators for the Toffoli gate, containing only single- and two-qubit interactions. This is a more detailed version of the analysis given in section 3.1.

The Toffoli gate can be written as

$$\mathcal{U}_{\rm Toff} = Z_1^+ + Z_1^-[Z_2^+ + Z_2^-(X_3^+ - X_3^-)], \tag{S7}$$

where we defined $Z_i^\pm \equiv (1 \pm Z_i)/2$, and similarly for $X_i^\pm$. The principal generator of $\mathcal{U}_{\rm Toff}$ is $\mathcal{H}_{\rm Toff} = \pi Z_1^- Z_2^- X_3^-$, that is, highlighting the three-qubit interaction term,

$$\mathcal{H}_{\rm Toff} = (\text{1- and 2-qubit terms}) - \pi/8\, Z_1 Z_2 X_3. \tag{S8}$$

We start by writing down the general parametrization of an Hamiltonian containing only one- and two-qubit interactions:

$$\tilde{\mathcal{H}}_{\rm Toff} = h_0 \mathbb{1} + \sum h_{i,\alpha}\sigma_i^\alpha + \sum J_{i,j}^{\alpha,\beta}\sigma_i^\alpha\sigma_j^\beta. \tag{S9}$$

Assuming for simplicity that $\tilde{\mathcal{H}}_{\rm Toff}$ does not contain Y_i components, and imposing the commutativity with $\mathcal{H}_{\rm Toff}$, we get the following expression

$$\begin{aligned}\tilde{\mathcal{H}}_{\rm Toff} =& h_0\mathbb{1} + h_3^x X_3 + h_1^z Z_1 + h_2^z Z_2 + J_{13}^{zx} Z_1 X_3 + J_{23}^{zx} Z_2 X_3 + J_{12}^{zz} Z_1 Z_2 \\ &+ (J_{13}^{xx} X_1 + J_{23}^{xx} X_2)(1 + X_3) + (J_{12}^{zx} X_2 + J_{13}^{zz} Z_3)(1 + Z_1) \\ &+ (J_{12}^{xz} X_1 + J_{23}^{zz} Z_3)(1 + Z_2).\end{aligned} \tag{S10}$$

As can be directly verified, the above satisfies $[\tilde{\mathcal{H}}_{\rm Toff}, \mathcal{H}_{\rm Toff}] = 0$ while also containing only one- and two-qubit interactions, and no Pauli Y matrices. We could now directly try and find the set of parameters making the eigenvalues of $\tilde{\mathcal{H}}_{\rm Toff} - \mathcal{H}_{\rm Toff}$ all integer multiples of $2\pi i$, but the associated calculations are made hard by the many parameters involved. However, it turns out that we can make further assumptions on the form of $\tilde{\mathcal{H}}_{\rm Toff}$ and still obtain a viable family of solutions. One of them, leading to a satisfying family of solutions, is $J_{12}^{xz} = J_{12}^{zx} = 0$, $J_{13}^{zx} = J_{23}^{zx} = \pi/8$, $J_{13}^{xx} = J_{23}^{xx}$, $J_{23}^{zz} = -J_{13}^{zz}$, and $h_1^z = h_2^z = -\pi/8$. With these assumptions equation (S10) becomes

$$\tilde{\mathcal{H}}_{\rm Toff} = h_0\mathbb{1} + h_3^x X_3 - \pi/8(Z_1 + Z_2)(1 - X_3) + J_{12}^{zz} Z_1 Z_2 + J_{13}^{xx}(X_1 + X_2)(1 + X_3) + J_{13}^{zz}(Z_1 - Z_2)Z_3. \tag{S11}$$

Finally, we impose that the generator is diagonal on the first two qubits, that is, $J_{13}^{xx} = 0$. Using this simplified expression, $\mathcal{H}'_{\rm Toff} = \tilde{\mathcal{H}}_{\rm Toff} - \mathcal{H}_{\rm Toff}$ becomes

$$\mathcal{H}'_{\rm Toff} = \pi/8\, Z_1 Z_2 X_3 + (h_0 - \pi/8)\mathbb{1} + (h_3^x + \pi/8)X_3 + (J_{12}^{zz} - \pi/8)Z_1 Z_2 + J_{13}^{zz}(Z_1 - Z_2)Z_3. \tag{S12}$$

With the above simplified expression it is now possible to directly solve the eigenvalue problem. This results in the following family of solutions

$$\begin{aligned}\tilde{\mathcal{H}}_{\rm Toff} = \frac{\pi}{8}\Big[&1 + 4\left(\nu_1 + \nu_2 + 2\nu_3 + \sqrt{(\nu_3 - \nu_4)^2}\right) - (Z_1 + Z_2)(1 - X_3) + X_3(-2 - 8\nu_1 + 8\nu_2) \\ &+ 4Z_1 Z_2\left(1/4 + \nu_1 + \nu_2 - 2\nu_3 - \sqrt{(\nu_3 - \nu_4)^2}\right) + (Z_2 - Z_1)Z_3\ \sqrt{c(\nu_1, \nu_2, \nu_3, \nu_4)}\Big],\end{aligned} \tag{S13}$$

with $c(\nu_1, \nu_2, \nu_3, \nu_4) = (4\nu_{34})^2 - (1 + 4\nu_{12})^2$, for all integer values of ν_i such that $c(\nu_1, \nu_2, \nu_3, \nu_4) \geqslant 0$. The corresponding spectrum of $\mathcal{H}'_{\rm Toff} = \tilde{\mathcal{H}}_{\rm Toff} - \mathcal{H}_{\rm Toff}$ is

$$\begin{aligned}\lambda_1 &= \lambda_2 = 2\pi\nu_1, \\ \lambda_3 &= \lambda_4 = 2\pi\nu_2, \\ \lambda_5 &= \lambda_6 = 2\pi\nu_3, \\ \lambda_7 &= \lambda_8 = 2\pi(\nu_3 + |\nu_3 - \nu_4|),\end{aligned} \tag{S14}$$

while the spectrum of $\tilde{\mathcal{H}}_{\rm Toff}$ changes only in that $\lambda_2 = 2\pi(\nu_1 + 1/2)$. Consistently with this, λ_2 is also the eigenvalue corresponding to the non-degenerate eigenspace of $\mathcal{H}_{\rm Toff}$, while all the other eigenvalues correspond to eigenvectors orthogonal to this one. More specifically, we have

$$\begin{aligned}|\lambda_1\rangle &= |0, 0, -\rangle, \quad |\lambda_2\rangle = |1, 1, -\rangle, \\ |\lambda_3\rangle &= |1, 1, +\rangle, \quad |\lambda_4\rangle = |0, 0, +\rangle, \\ |\lambda_5\rangle &= |1, 0\rangle \otimes N_5\left[(a - b)|0\rangle + |1\rangle\right], \\ |\lambda_6\rangle &= |0, 1\rangle \otimes N_6\left[(a + b)|0\rangle + |1\rangle\right], \\ |\lambda_7\rangle &= |1, 0\rangle \otimes N_6\left[(a + b)|0\rangle - |1\rangle\right], \\ |\lambda_8\rangle &= |0, 1\rangle \otimes N_5\left[(a - b)|0\rangle - |1\rangle\right],\end{aligned} \tag{S15}$$

Table A1. Toffoli.

$$P_1 = Z_1^+ Z_2^+ X_3^-, \quad P_2 = Z_1^- Z_2^- X_3^-, \quad P_3 = Z_1^+ Z_2^+ X_3^+, \quad P_4 = Z_1^- Z_2^- X_3^+,$$
$$P_5 = Z_1^- Z_2^+ \frac{1}{2|\bar\nu_{34}|}\left[|\bar\nu_{34}| + (1+\bar\nu_{12})X_3 - \sqrt{-(1+\bar\nu_{12})^2 + \bar\nu_{34}^2}Z_3\right],$$
$$P_6 = Z_1^+ Z_2^- \frac{1}{2|\bar\nu_{34}|}\left[|\bar\nu_{34}| + (1+\bar\nu_{12})X_3 + \sqrt{-(1+\bar\nu_{12})^2 + \bar\nu_{34}^2}Z_3\right],$$
$$P_7 = Z_1^- Z_2^+ \frac{1}{2|\bar\nu_{34}|}\left[|\bar\nu_{34}| - (1+\bar\nu_{12})X_3 + \sqrt{-(1+\bar\nu_{12})^2 + \bar\nu_{34}^2}Z_3\right],$$
$$P_8 = Z_1^+ Z_2^- \frac{1}{2|\bar\nu_{34}|}\left[|\bar\nu_{34}| - (1+\bar\nu_{12})X_3 - \sqrt{-(1+\bar\nu_{12})^2 + \bar\nu_{34}^2}Z_3\right].$$

$$P_1 + P_2 = \frac{1}{4}(1+Z_1Z_2)(1-X_3), \quad P_3 + P_4 = \frac{1}{4}(1+Z_1Z_2)(1+X_3).$$
$$P_5 + P_6 = \frac{1}{4|\bar\nu_{34}|}\left[(1-Z_1Z_2)|\bar\nu_{34}| + (1-Z_1Z_2)X_3(1+\bar\nu_{12}) + (Z_1 - Z_2)Z_3\sqrt{\bar\nu_{34}^2 - (1+\bar\nu_{12})^2}\right],$$
$$P_7 + P_8 = \frac{1}{4|\bar\nu_{34}|}\left[(1-Z_1Z_2)|\bar\nu_{34}| - (1-Z_1Z_2)X_3(1+\bar\nu_{12}) - (Z_1 - Z_2)Z_3\sqrt{\bar\nu_{34}^2 - (1+\bar\nu_{12})^2}\right].$$

It is easily verified from the above that

$$P_1 + P_2 + P_3 + P_4 = \tfrac{1}{2}(1+Z_1Z_2), \quad P_5 + P_6 + P_7 + P_8 = \tfrac{1}{2}(1-Z_1Z_2),$$

so that the sum of the projectors gives the identity as it should.On the other hand, multiplying by the appropriate ν_i factors, we get

$$2\pi\left[\nu_1(P_1+P_2) + \nu_2(P_3+P_4)\right] = (\ldots) + \frac{\pi}{2}(\nu_2 - \nu_1)Z_1Z_2X_3,$$
$$2\pi\left[\nu_3(P_5+P_6) + \nu_4(P_7+P_8)\right] = (\ldots) + \frac{\pi}{2}(\nu_1 - \nu_2)Z_1Z_2X_3 + \frac{\pi}{8}Z_1Z_2X_3,$$

with the last identity holding for $\nu_3 \neq \nu_4$.

with

$$a = \frac{|\bar\nu_{34}|}{1+\bar\nu_{12}}, \quad b = \frac{\sqrt{\bar\nu_{34}^2 - (\bar\nu_{12}+1)^2}}{1+\bar\nu_{12}}, \tag{S16}$$

It is worth noting that the orthogonality of these eigenvectors follows from the easily verified property of the above coefficients: $a^2 - b^2 = 1$. Furthermore, we note that $c(\nu_1, \nu_2, \nu_3, \nu_4) \geqslant 0$ cannot be satisfied unless $\nu_3 \neq \nu_4$. This in turn, looking at equation (S15), reveals that all the solutions are made possible by a non-trivial lifting of the degeneracy of the subspaces $|0,1\rangle\langle 0,1|$ and $|1,0\rangle\langle 1,0|$. Let us now try to understand how and why the derived $\mathcal{H}'_{\text{Toff}}$ works. Let us use the notation $P_i \equiv |\lambda_i\rangle\langle\lambda_i|$, and consider the projector over the last two eigenvectors. Highlighting the three-qubit terms, we find

$$\begin{aligned} P_7 &\simeq -N_6^2 \frac{Z_1Z_2}{4}\left[\left((a+b)^2 - 1\right)\frac{Z_3}{2} - (a+b)X_3\right], \\ P_8 &\simeq -N_5^2 \frac{Z_1Z_2}{4}\left[\left((a-b)^2 - 1\right)\frac{Z_3}{2} - (a-b)X_3\right]. \end{aligned} \tag{S17}$$

The term in the Hamiltonian to which these two projectors contribute is $2\pi\nu_{3,4}(P_7 + P_8)$, with $\nu_{3,4} = \nu_3 + |\nu_3 - \nu_4|$. Recalling the definitions of a, b, N_5, N_6, we see that the coefficient of $Z_1Z_2Z_3$ vanishes, and the resulting expression becomes

$$P_7 + P_8 \simeq Z_1Z_2X_3\frac{1+4(\nu_1 - \nu_2)}{16|\nu_3 - \nu_4|}. \tag{S18}$$

Substitution of the appropriate values of ν_i shows that the above term can be used to generate the three-qubit factor $\pi/8\, Z_1\, Z_2\, X_3$, *without introducing additional three-qubit factors*. In table A1 are given the full expressions for the projectors and the found solutions for the Toffoli gate. It is also interesting to note that all of the above still holds if the X_i operators are replaced with Y_i operators. That is, the expressions found solving for the Toffoli, by simple substitution $X_i \to Y_i$, also give a generator with only two-qubit interactions for the CCY gate (that is, the gate that applies Y to the third qubit conditionally to the first two qubits being in the $|1\rangle$ state).

A different way to understand $\tilde{\mathcal{H}}_{\mathrm{Toff}}$ is to analyse the four two-dimensional subspaces on the main diagonal, exploiting the fact that $\tilde{\mathcal{H}}_{\mathrm{Toff}}$ acts diagonally on the first two qubits. Straightforward calculations lead to

$$\begin{aligned}
\langle 00|\tilde{\mathcal{H}}_{\mathrm{Toff}}|00\rangle &= \pi\left[(\nu_1+\nu_2)-(\nu_1-\nu_2)X\right],\\
\langle 01|\tilde{\mathcal{H}}_{\mathrm{Toff}}|01\rangle &= 2\pi\nu_3+\pi|\nu_{34}|(1-\sigma_{01}),\\
\langle 10|\tilde{\mathcal{H}}_{\mathrm{Toff}}|10\rangle &= 2\pi\nu_3+\pi|\nu_{34}|(1-\sigma_{10}),\\
\langle 11|\tilde{\mathcal{H}}_{\mathrm{Toff}}|11\rangle &= \frac{\pi}{2}\left[(1+2(\nu_1+\nu_2))-(1+2\nu_{12})X\right],
\end{aligned}$$

where

$$\sigma_{01}\equiv\frac{(1+4\nu_{12})X+\sqrt{c}Z}{4|\nu_{34}|},\qquad \sigma_{10}\equiv\frac{(1+4\nu_{12})X-\sqrt{c}Z}{4|\nu_{34}|}. \tag{S19}$$

It can be verified that for all values of ν_1,ν_2,ν_3,ν_4, the two-dimensional identity and X are correctly generated in the $|00\rangle$ and $|11\rangle$ spaces, respectively. On the other hand, in the $|01\rangle$ and $|10\rangle$ spaces, the two-dimensional identity is generated as long as $\nu_3\neq\nu_4$, as was also derived before. In particular, the class of solutions given by $\nu_1=\nu_2=\nu_3=0$ is

$$\tilde{\mathcal{H}}=\frac{\pi}{8}\left[1+4|\nu_4|-2X_3-Z_1-Z_2+(Z_1+Z_2)X_3+Z_1Z_2(1-4|\nu_4|)+(Z_2-Z_1)Z_3\sqrt{16\nu_4^2-1}\right], \tag{S20}$$

for all $\nu_4\neq 0$. It is interesting to look at the explicit form of the exponentials generated by this class generators. Computing using equation (S20), we get

$$\exp(\mathrm{i}\tilde{\mathcal{H}}t)=\begin{pmatrix}\mathbb{1}_2 & \mathbb{0} & \mathbb{0} & \mathbb{0}\\ \mathbb{0} & S(t,\nu_4) & \mathbb{0} & \mathbb{0}\\ \mathbb{0} & \mathbb{0} & S(t,\nu_4) & \mathbb{0}\\ \mathbb{0} & \mathbb{0} & \mathbb{0} & X(t)\end{pmatrix}, \tag{S21}$$

where

$$S(t,\nu_4)=\begin{pmatrix}a+b & c\\ c & a-b\end{pmatrix}, \tag{S22}$$

$$\begin{aligned}
a&=\frac{1+\mathrm{e}^{2\mathrm{i}\pi t\nu_4}}{2},\qquad c=\frac{1-\mathrm{e}^{2\mathrm{i}\pi t\nu_4}}{8\nu_4},\\
b&=\frac{(-1+\mathrm{e}^{2\mathrm{i}\pi t\nu_4})\sqrt{16\nu_4^2-1}}{8\nu_4},
\end{aligned} \tag{S23}$$

and

$$X(t)=\frac{1}{2}\begin{pmatrix}(1+\mathrm{e}^{\mathrm{i}\pi t}) & (1-\mathrm{e}^{\mathrm{i}\pi t})\\ (1-\mathrm{e}^{\mathrm{i}\pi t}) & (1+\mathrm{e}^{\mathrm{i}\pi t})\end{pmatrix}. \tag{S24}$$

For large (in modulus) values of ν_4, $a+b\to\mathrm{e}^{2\mathrm{i}\pi t\nu_4}$, $a-b\to 1$ and $c\to 0$, so that the exponential becomes

$$\exp(\mathrm{i}\tilde{\mathcal{H}}t)=\begin{pmatrix}\mathbb{1}_2 &&&&&\\ & \mathrm{e}^{2\mathrm{i}\pi t\nu_4} &&&&\\ && 1 &&&\\ &&& \mathrm{e}^{2\mathrm{i}\pi t\nu_4} &&\\ &&&& 1 &\\ &&&&& X(t)\end{pmatrix}, \tag{S25}$$

which very closely resembles the matrix obtained by exponentiating the principal generator $\mathcal{H}_{\mathrm{Toff}}=\pi Z_1^- Z_2^- X_3^-$, that is

$$\exp(\mathrm{i}\mathcal{H}_{\mathrm{Toff}}t)=\begin{pmatrix}\mathbb{1}_2 &&&\\ & \mathbb{1}_2 &&\\ && \mathbb{1}_2 &\\ &&& X(t)\end{pmatrix}. \tag{S26}$$

A different solution derived from equation (S13) is

$$\tilde{\mathcal{H}}_{\mathrm{Toff}}=\frac{9\pi}{8}+\frac{3\pi}{4}X_3-\frac{\pi}{8}(Z_1+Z_2)+\frac{\pi}{8}Z_1Z_2+\frac{\pi}{8}(Z_1+Z_2)X_3-\frac{\sqrt{7}\pi}{8}(Z_1-Z_2)Z_3. \tag{S27}$$

Moreover, it is worth noting that equation (S13) is only one possible family of solutions, and that different assumptions will lead to different ones. For example, a similar reasoning as above, starting however from the assumptions $J^{zz}_{23} = J^{zz}_{13}$ will lead to solutions such as [note the use of $(Z_1 + Z_2)$ terms here, making this solution not derivable from equation (S13)]

$$\begin{aligned}\tilde{\mathcal{H}}_{\text{Toff}} =& \frac{9\pi}{8} - \frac{7\pi}{8}X_3 + \frac{\sqrt{15}\pi}{8}Z_3 + \frac{\pi}{8}Z_1Z_2 \\ &+ \frac{\pi}{8}(Z_1+Z_2)\left(-1 + \frac{5}{2}X_3 + \frac{\sqrt{15}}{2}Z_3\right).\end{aligned} \tag{S28}$$

A4. Toffoli gate: an example of direct *a posteriori* derivation

We will here show a line of thought that could have conceivably led to equation (S20) (in the case $\nu_4 = 1$), by direct analysis, and without using any of the tools shown in the paper. It will be useful to keep in mind the expressions of $Z_1 \pm Z_2$:

$$\begin{aligned} Z_2 + Z_1 &= \text{diag}(2,2,0,0,0,0,-2,-2), \\ Z_2 - Z_1 &= \text{diag}(0,0,-2,-2,2,2,0,0). \end{aligned} \tag{S29}$$

Given that we want to generate a CC-X gate, and remembering that $\exp\left[\frac{i\pi}{2}(1-X)\right] = X$, it is reasonable to start building our Hamiltonian as

$$\mathcal{H}_1 = -\pi\left(\frac{Z_1+Z_2}{2}\right)\left(\frac{1-X_3}{2}\right), \tag{S30}$$

which however will generate an X evolution both in the $|00\rangle$ and in the $|11\rangle$ sectors, while we want it only in the latter sector: $\mathcal{H}_1 \doteq \text{diag}(-X^-, 0, 0, X^-)$. We can remove the term in the $|00\rangle$ sector exploiting the sign difference introduced by $Z_1 + Z_2$, by directly adding an appropriate one-qubit interaction term:

$$\begin{aligned}\mathcal{H}_2 &= \frac{1}{2}\left[\mathcal{H}_1 + \frac{\pi}{2}(1-X_3)\right] \\ &= \pi\ \text{diag}(0, X^-/2, X^-/2, X^-),\end{aligned} \tag{S31}$$

with $\exp(i\pi X^-) = X$. Equation (S31) now correctly reproduces the evolution on $|00\rangle$ and $|11\rangle$, but also wrongly evolves $|01\rangle$ and $|10\rangle$. To remove these additional terms while at the same time leaving the others unaffected we use the fact that $\exp(i\pi(1\pm\boldsymbol{\sigma})) = \mathbb{1}$, for any normalized vector of sigma matrices: $\boldsymbol{\sigma} \equiv \sum_{i=1}^3 n_i\sigma_i$ with $n_1^2 + n_2^2 + n_3^2 = 1$. To convert the central terms in equation (S31) into something like this we observe that we can rewrite the second term in the above equation as

$$\pi/4(1-X_3) = \pi/8(2-2X_3) = \pi/8(5-3-2X_3). \tag{S32}$$

As $Z_1Z_2 = \text{diag}(1,-1,-1,1)$, we substitute the above with $\pi/8(5 - 3Z_1Z_2 - 2X_3)$. This change affects only the central terms, converting the expression into: $\pi\text{diag}(0, 1 - X/4, 1 - X/4, X^-)$. The reason this form is preferable is that we can now simply add a factor in the central terms to convert them into an expression of the form $1 - \boldsymbol{\sigma}$. Adding an interaction of the form $\pi\alpha(Z_2 - Z_1)Z_3$ gives

$$\pi\ \text{diag}(0, 1 - X/4 - 2\alpha Z, 1 - X/4 + 2\alpha Z, X^-). \tag{S33}$$

For the central terms to exponentiate to the identity we need them to become of the form $1 - \boldsymbol{\sigma}$ with normalized $\boldsymbol{\sigma}$. This is easily achieved by choosing $\alpha = \pm\sqrt{15}/8$. The final expression is thus

$$8/\pi\ \mathcal{H}_3 = -(Z_1+Z_2)(1-X_3) \pm \sqrt{15}(Z_2-Z_1)Z_3 + (5 - 3Z_1Z_2 - 2X_3). \tag{S34}$$

Note that instead of $\pi\alpha(Z_2 - Z_1)Z_3$ we could have equivalently chosen $\pi\alpha(Z_2 - Z_1)O_3$ for any $O_3 = aY_3 + bZ_3$ and $a^2 + b^2 = 1$. The above reasoning explains the origin of the weird $\sqrt{15}$ factor: it comes as the coordinate necessary to make the vector unitary: for $X/4 + xZ/4$ to be normalized, $x = \sqrt{15}$ must be satisfied.

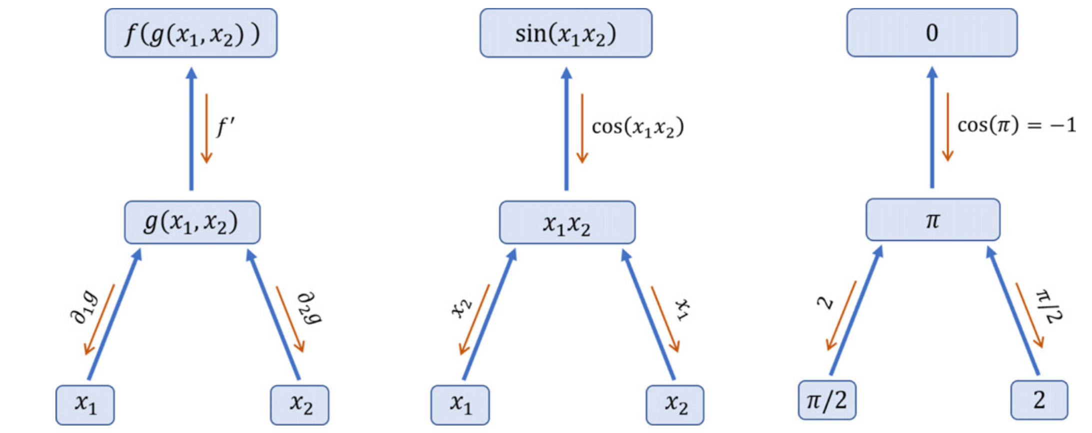

Figure B1. Examples of AD in backpropagation mode. (a) Schematic representation of AD of a function with one output and two inputs. Starting from numerical values for x_1 and x_2, one computes $g(x_1, x_2)$ and then $f(g(x_1, x_2))$. To get $\nabla f(g(x_1, x_2))$, one then computes $f'(g(x_1, x_2))\partial_i g(x_1, x_2)$. Note that all components of this expression are known: f' and $\partial_i g$ are known by assumption, and the value of $g(x_1, x_2)$ has been computed and cached in the forward propagation phase. (b) Example of application of AD to compute the gradient of $\cos(x_1 x_2)$. (c) Using the same example function as (b), we give an example of the actual number computed at all stages when the inputs are $(x_1, x_2) = (\pi/2, 2)$.

Appendix B. Supervised learning approach

We here study more in depth the following problem: given a target gate $\mathcal{G}$ and a parametrized Hamiltonian $\mathcal{H}(\boldsymbol{\lambda}) = \sum_k \lambda_k \sigma_k$, with $\lambda_k \in \mathbb{R}$ and σ_k Hermitian operators, what is the set of parameters $\boldsymbol{\lambda}_0$ such that $\exp(\mathrm{i}\mathcal{H}(\boldsymbol{\lambda}_0)) = \mathcal{G}$? We present a supervised learning approach to numerically solve this problem, considerably extending the ideas presented in reference [16]. Thanks to a number of numerical optimization techniques, and in particular the use of AD [20–23], we can explore a variety of different scenarios, optimizing over potentially hundreds of Hamiltonian parameters. On top of this, condition (1b) is used to further speed-up the numerical training, removing many interaction parameters that are known not to lead to the target gate.

B1. Supervised learning

Supervised learning is the task of inferring or approximating a function, given a set of pre-labeled data [18, 49]. A supervised learning algorithm starts with some *model*—a functional relation $g_{\boldsymbol{\lambda}}$ parametrized by a set of parameters $\boldsymbol{\lambda}$—and finds a $\boldsymbol{\lambda}_0$ making $g_{\boldsymbol{\lambda}_0}$ as close as possible to a target function f. To do this, a set of pre-labeled *training data* $\{(x_1, y_1), (x_2, y_2), \ldots\}$ is used, where here $y_k = f(x_k)$ is the output that we want the algorithm to associate to the input x_k.

Among the most used supervised learning models are neural networks (NNs) [50, 51]. These are parametric non-linear models which play a prominent role in many machine learning tasks, such as dimensionality reduction, classification, and feature extraction [50, 51]. NNs have also recently proven useful for several problems in quantum many-body theory [52–59], quantum compilation [60], quantum stabilizer codes [61] and entanglement quantification [62].

A NN is *trained* by optimizing its parameters using a dataset of pre-labeled data. A common way to do this is use variations of a gradient-descent-based technique named SGD. Gradient descent algorithms aim to optimize a problem function $f(\boldsymbol{x})$, starting from an initial point $\boldsymbol{x}_0$ and performing a number of small steps towards the direction of maximum slope (that is, $\nabla f(\boldsymbol{x})$). The optimal point $\boldsymbol{x}_{\text{opt}}$ is thus obtained via a sequence of small perturbations of the point $\boldsymbol{x}$, which starting from $\boldsymbol{x}_0$ reaches the nearest local stationary point by following the slope. In the simplest version of the algorithm, the update rule is simply $\boldsymbol{x} \to \boldsymbol{x} - \eta\nabla f(\boldsymbol{x})$, with η a small real parameter commonly referred to as *learning rate*. SGD, on the other hand, is suitable for a situation in which one is given a parametrized functional relationship of the form $f(\boldsymbol{x}; \boldsymbol{w})$, and asked for a set of 'parameters' $\boldsymbol{w}_0$ such that $f(\boldsymbol{x}; \boldsymbol{w}_0)$ is minimum (maximum) for all *inputs* $\boldsymbol{x}$. Such a case can be handled via SGD, which in its simplest form involves picking a random $\boldsymbol{x}_1$, executing a number of gradient descent iterations over $\boldsymbol{w}$, then picking a new $\boldsymbol{x}_2$ and iterating the procedure. The updating rule for SGD is therefore of the form

$$\boldsymbol{w} \to \boldsymbol{w} - \eta\nabla_{\boldsymbol{w}} f(\boldsymbol{x}; \boldsymbol{w}). \tag{S35}$$

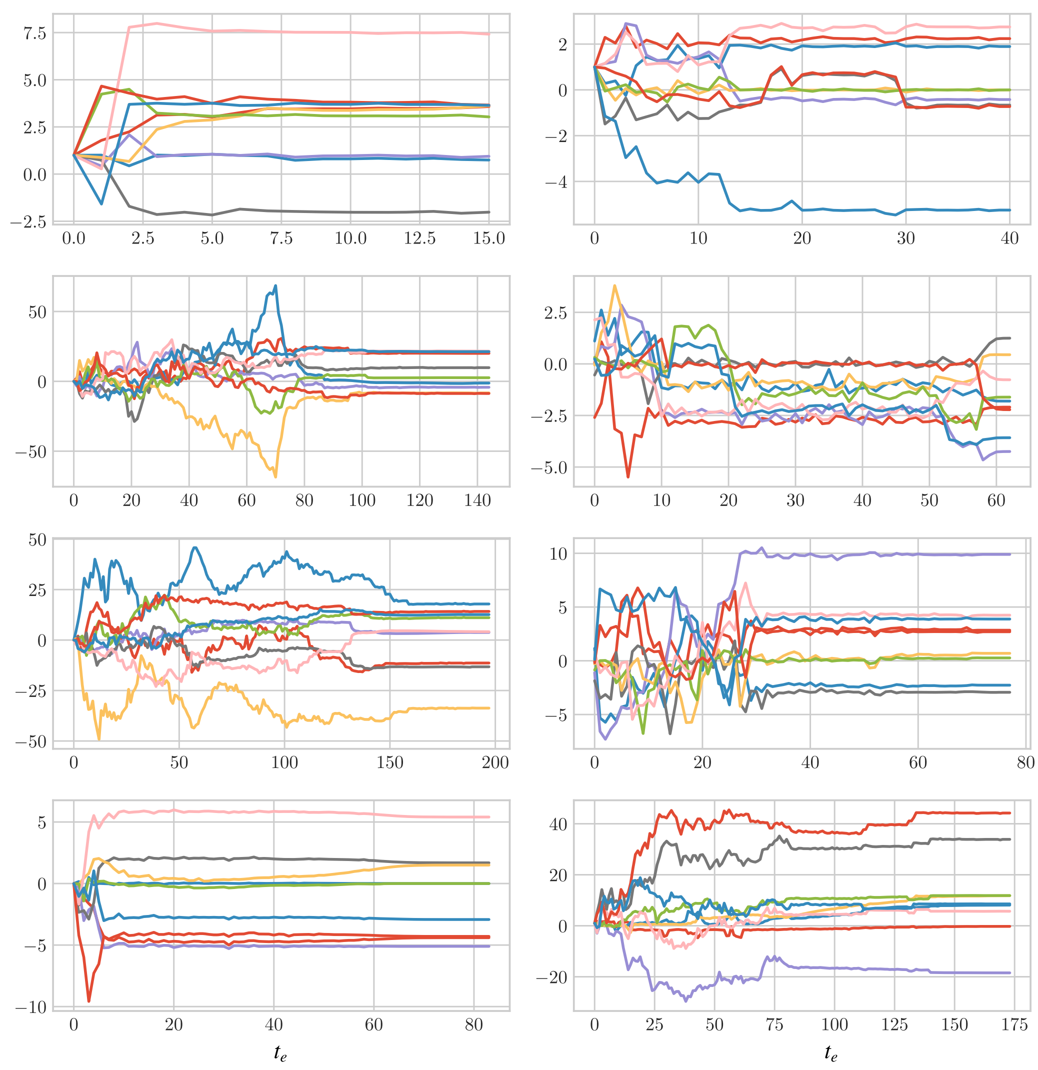

Figure B2. Training histories for the Toffoli gate with only diagonal interactions. In each plot are reported the values of the nine network parameters during the training process, for each training epoch t_e. Each training process was left running until convergence to unit fidelity, therefore, the number of epochs in the horizontal axes differs for different trainings instances. The histories shown here correspond to training instances in which the parameters were initialised at various values, as seen from the leftmost values in each plot.

While standard gradient descent, being a local optimization algorithm, is liable to getting stuck in local minima, SGD can at least partially avoid this issue, in that generally a local minimum for an input $\boldsymbol{x}$ is not a local minimum for a different input $\boldsymbol{x}'$. Many variations of SGD are used in different circumstances. For example, in the so-called *mini-batch* SGD, instead of updating with a single input $\boldsymbol{x}$, one uses a *batch* of inputs $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_M\}$, and updates the parameters using the averaged gradient: $\boldsymbol{w} \to \boldsymbol{w} - \eta \sum_{k=1}^{M} \nabla_{\boldsymbol{w}} f(\boldsymbol{x}_k; \boldsymbol{w})/M$. More sophisticated updating rules are used to increase the training efficiency in different circumstances. Common techniques involve dynamically updating the learning rate, or using *momentum gradient descent* [37, 41] techniques.

To see how this class of optimization problems is relevant to us, consider the fidelity function $\mathcal{F}$ defined as

$$\mathcal{F}(\boldsymbol{\lambda}, \psi) \equiv \langle \psi | \mathcal{G}^\dagger \exp(\mathrm{i}\mathcal{H}(\boldsymbol{\lambda}))\mathcal{G} | \psi \rangle, \tag{S36}$$

with $\mathcal{G}$ the target gate, $\boldsymbol{\lambda}$ the set of parameters, and ψ an input state. The gate design problem is then equivalent to finding $\boldsymbol{\lambda}$ such that $\mathcal{F}(\boldsymbol{\lambda}, \psi)$ is maximised (that is, equal to 1) for all ψ. One possibility to solve this problem is to consider the average fidelity $\bar{\mathcal{F}}(\boldsymbol{\lambda})$, for which explicit formulas are known [63–65]. Standard optimization methods, like standard gradient descent or differential evolution, can be applied directly on $\bar{\mathcal{F}}(\boldsymbol{\lambda})$. This, however, reveals to be impractical, due to the complexity of the associated parameter landscapes. On the other hand, SGD allows to use a simple and efficient local maximisation

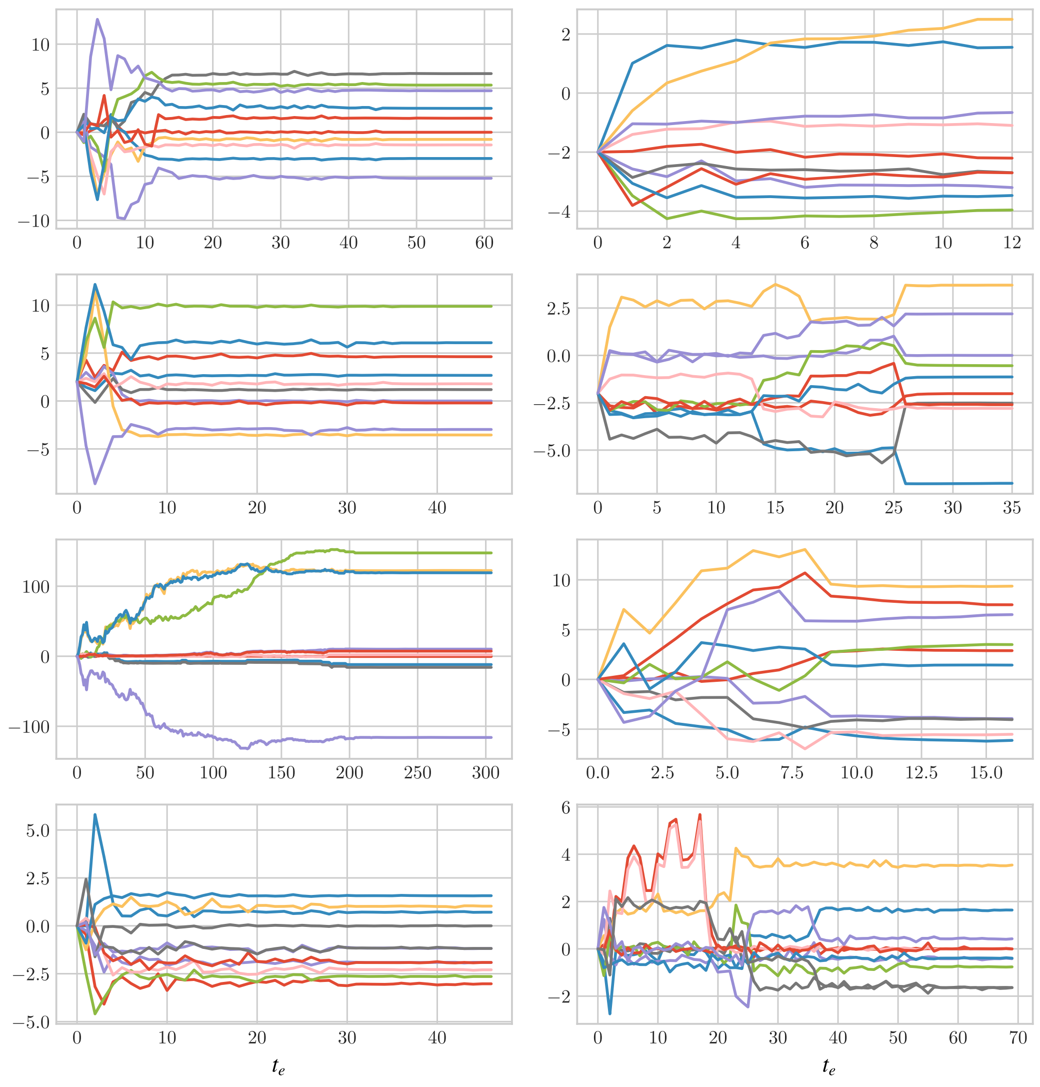

Figure B3. Training histories for the Fredkin gate with only diagonal interactions. In each plot are reported the values of the nine network parameters during the training process, for each training epoch t_e. Each training process was left running until convergence to unit fidelity, therefore, the number of epochs in the horizontal axes differs for different trainings instances. The histories shown here correspond to training instances in which the parameters were initialised at various values, as seen from the leftmost values in each plot.

method, while at the same time being less prone to getting stuck in local maxima. This works particularly well in this case, because we know that the sets of parameters corresponding to the target gate are *all and only* those such that *for all inputs* ψ the fidelity is unitary. Despite this, the supervised learning approach becomes more efficient when more states are used: using only states belonging to a basis would allow the optimizer to only operate on a small number of curves. Using an overcomplete set of states makes the algorithm much more robust and efficient.

A crucial step, efficiency-wise, in gradient descent algorithms, is the evaluation of the gradient. Numerically approximating the gradient, as done in previous works [16], is generally inefficient and scales badly with the number of optimized parameters. Here we will instead make use of the powerful technique of automatic differentiation (AD) [21, 23], described in appendix B2. AD dramatically improves the training efficiency, allowing to explore a richer variety of circumstances.

B2. Backpropagation

The gradient evaluation phase is efficiency-wise crucial for the training of a neural network. Computing the partial derivatives of the cost function with a standard method, like finite differences, has a complexity $\mathcal{O}(N_w^3)$, with N_w the number of parameters to differentiate [18]. This inefficiency can however be avoided

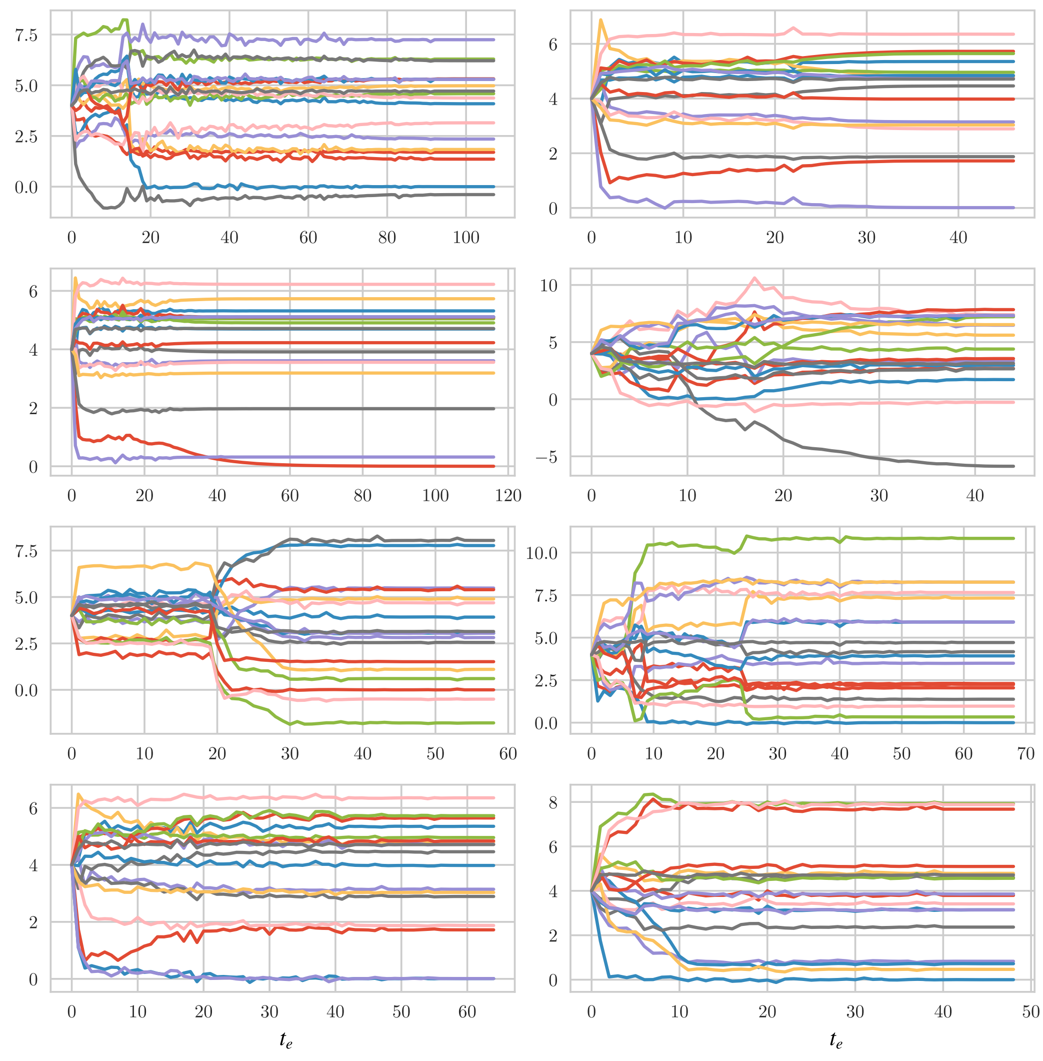

Figure B4. Training histories for the double-Fredkin gate with only diagonal interactions. In each plot are reported the values of the 18 network parameters during the training process, for each training epoch t_e. Each training process was left running until convergence to unit fidelity, therefore, the number of epochs in the horizontal axes differs for different trainings instances. All the histories shown here correspond to training instances in which the parameters were initialised to four.

using *error backpropagation* via AD. With this technique, the complexity of the gradient evaluation phase can be cut down to $\mathcal{O}(N_{\boldsymbol{w}}^2)$ [18]. This works by first decomposing the cost function of the model in terms of elementary operations, that is, functions the gradient of which is known analytically. In this way the *computational graph* representing the functional relation between input and output is built. A computational graph is a directed acyclic graph, whose nodes represent the operations, and edges the flowing direction of inputs into outputs (see figure B1). Once the computational graph is built, the gradients with respect to the model parameters can be computed efficiently. This happens in two stages, as schematically illustrated in figure B1. At first, every node of the computational graph is progressively computed, starting from the inputs (the current values of the model parameters) up to the final value of the error function. During this process, the intermediate values of the elementary operations are cached. This is the so-called *feed-forward* phase. The second phase (so-called *backpropagation* phase) starts from the output, and consists of progressively computing the gradients of the error function with respect to the independent variables.

To better understand AD, let us consider a simple example. Suppose the error function of the model is of the form $g(\boldsymbol{w}) \equiv f(\boldsymbol{f}^{(2)}(\boldsymbol{f}^{(1)}(\boldsymbol{w})))$, where $\boldsymbol{w}$ is a set of parameters, and $\boldsymbol{f}^{(i)}$ are intermediate 'elementary' functions, the gradients of which are supposed to be known analytically. Making use of the chain rule, the gradient of g reads

$$\nabla g(\boldsymbol{w}) = \sum_k \partial_k f(\boldsymbol{y}^{(2)}) \nabla f_k^{(2)}(\boldsymbol{y}^{(1)}), \tag{S37}$$

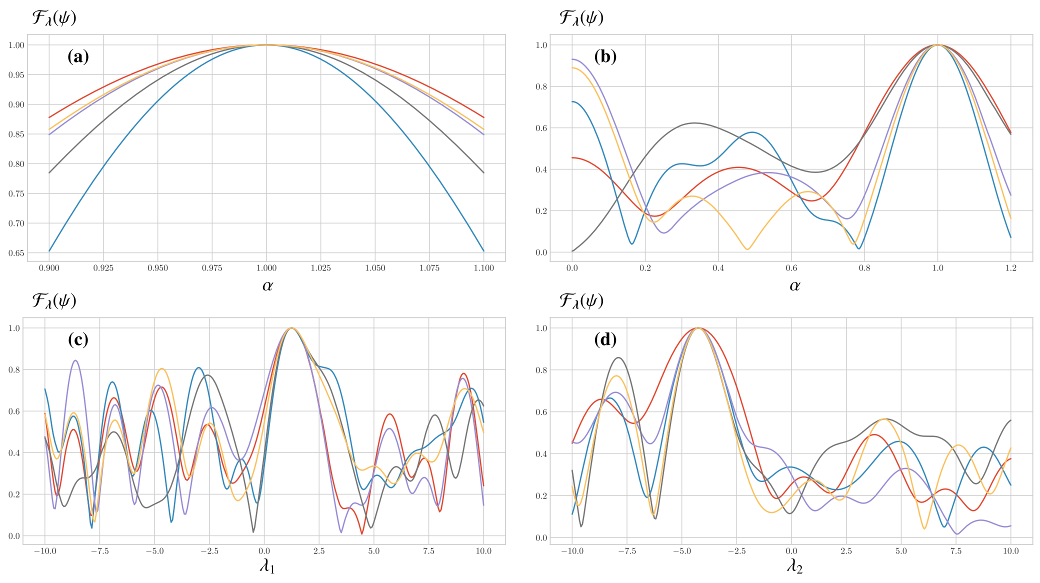

Figure B5. Fidelity $\mathcal{F}_{\boldsymbol{\lambda}}(\psi)$ vs variations of $\boldsymbol{\lambda}$, for different test states, for the **Toffoli** gate. The five test states ψ are sampled randomly. (a) Global relative variations of $\boldsymbol{\lambda}$, that is, plotting the fidelity against $\alpha\boldsymbol{\lambda}$ for $0.9 \leqslant \alpha \leqslant 1.1$. Note that this is equivalent to studying how the fidelity changes with respect to uncertainties in the evolution time, that is, how much does $\exp(\mathrm{i}\mathcal{H}t')$ differ from $\exp(\mathrm{i}\mathcal{H}t)$. (b) Same as (a) but with $0 \leqslant \alpha \leqslant 1.2$. (c) Plot of $\mathcal{F}_{\boldsymbol{\lambda}}(\psi)$ against *absolute* variations of a single element of $\boldsymbol{\lambda}$, in this case the first one, i.e. we take $\lambda_1 \in [-10, 10]$. (d) Like (c) but for λ_2.

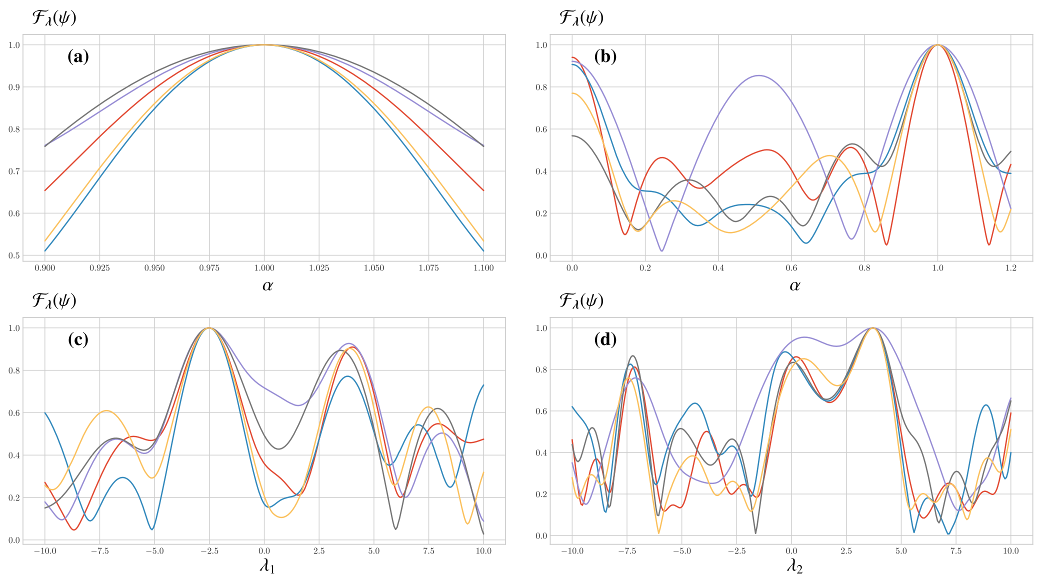

Figure B6. Fidelity $\mathcal{F}_{\boldsymbol{\lambda}}(\psi)$ vs variations of $\boldsymbol{\lambda}$, for different test states, for the **Fredkin** gate. The five test states ψ are sampled randomly. (a) Global relative variations of $\boldsymbol{\lambda}$, that is, plotting the fidelity against $\alpha\boldsymbol{\lambda}$ for $0.9 \leqslant \alpha \leqslant 1.1$. Note that this is equivalent to studying how the fidelity changes with respect to uncertainties in the evolution time, that is, how much does $\exp(\mathrm{i}\mathcal{H}t')$ differ from $\exp(\mathrm{i}\mathcal{H}t)$. (b) Same as (a) but with $0 \leqslant \alpha \leqslant 1.2$. (c) Plot of $\mathcal{F}_{\boldsymbol{\lambda}}(\psi)$ against *absolute* variations of a single element of $\boldsymbol{\lambda}$, in this case the first one, i.e. we take $\lambda_1 \in [-10, 10]$. (d) Like (c) but for λ_2.

where $\boldsymbol{y}^{(2)} = \boldsymbol{f}^{(2)}(\boldsymbol{f}^{(1)}(\boldsymbol{w}))$ and $\boldsymbol{y}^{(1)} = \boldsymbol{f}^{(1)}(\boldsymbol{w})$. During the feed-forward phase the values of $\boldsymbol{y}^{(1)}$ and then $\boldsymbol{y}^{(2)}$ are progressively computed and cached. Using $\boldsymbol{y}^{(2)}$, and the known expression for $\partial_k f$, $\partial_k f(\boldsymbol{y}^{(2)})$ is then efficiently computed. The process continues by evaluating $\nabla f_k^{(2)}$, which is written as

$$\nabla f_k^{(2)}(\boldsymbol{y}^{(1)}) = \sum_j \partial_j f_k^{(2)}(\boldsymbol{y}^{(1)}) \nabla f_j^{(1)}(\boldsymbol{w}). \tag{S38}$$

Again, being $\boldsymbol{y}^{(1)}$ already computed during the feed-forward, $\partial_j f_k^{(2)}(\boldsymbol{y}^{(1)})$ is readily computed. The last component needed for the full gradient is $\partial_i f_j^{(1)}(\boldsymbol{w})$, all parts of which are known. This method therefore

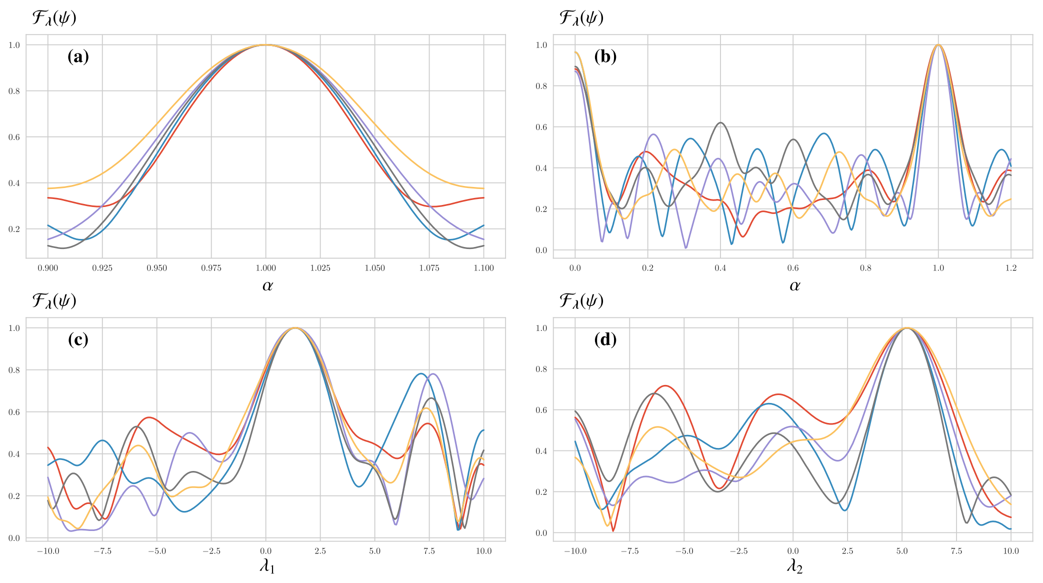

Figure B7. Fidelity $\mathcal{F}_{\boldsymbol{\lambda}}(\psi)$ vs variations of $\boldsymbol{\lambda}$, for different test states, for the **double Fredkin** gate. The five test states ψ are sampled randomly. (a) Global relative variations of $\boldsymbol{\lambda}$, that is, plotting the fidelity against $\alpha\boldsymbol{\lambda}$ for $0.9 \leqslant \alpha \leqslant 1.1$. Note that this is equivalent to studying how the fidelity changes with respect to uncertainties in the evolution time, that is, how much does $\exp(i\mathcal{H}t')$ differ from $\exp(i\mathcal{H}t)$. (b) Same as (a) but with $0 \leqslant \alpha \leqslant 1.2$. (c) Plot of $\mathcal{F}_{\boldsymbol{\lambda}}(\psi)$ against *absolute* variations of a single element of $\boldsymbol{\lambda}$, in this case the first one, i.e. we take $\lambda_1 \in [-10, 10]$. (d) Like (c) but for λ_2.

allows to efficiently evaluate numerical the gradient of complicated functions, without approximating the derivatives.

In the context of training neural networks, the function to be derived is the *cost function* of the network, that is, roughly speaking, the (Euclidean) distance between the result obtained for an input and the corresponding training output. For the gate design problem, we will use another notion of *distance* between output obtained and output expected. For quantum states, the fidelity between these turns out to work well.

B3. Implementation details

We used Python as language of choice for the implementation of the supervised learning. Being Python language of widespread use in the machine learning community, many libraries and frameworks are available to build computational graphs over which AD can be used. In particular, we used Theano [38], together with the QuTiP library for the simulation of the dynamics of quantum systems [66, 67].

Our implementation allows the training of an arbitrary target gate, parametrized via a time-independent Hamiltonian $\mathcal{H}(\boldsymbol{\lambda})$. The parametrization is completely arbitrary (provided the dependence on the parameters is linear), so that the Hamiltonian can be chosen as $\mathcal{H}(\boldsymbol{\lambda}) = \sum_i \lambda_i A_i$ for any set of matrices A_i and number of parameters λ_i. This is made possible by the flexibility of AD, which allows to automatically build an efficiently differentiable computational graph, without needing to hardcode the structure of the Hamiltonian.

The goal of the algorithm is, given a target gate $\mathcal{G}$ and a parametrization for the Hamiltonian $\mathcal{H}(\boldsymbol{\lambda})$, find the $\boldsymbol{\lambda}_0$ such that $\exp(i\mathcal{H}(\boldsymbol{\lambda}_0)) = \mathcal{G}$. We use for the purpose mini-batch SGD with momentum. The *mini-batch* version of SGD involves computing the gradient, at every iteration, averaging over the gradients computed for a number of states. Making such *batches* of states larger or smaller allows to enhance or decrease the variance of the gradients with respect to the input state. The use of *momentum* [37, 41] involves using a modified version of equation (S1). The updating rule is instead given by

$$\begin{aligned} \boldsymbol{v} &\to \gamma \boldsymbol{v} + \eta \nabla_{\boldsymbol{\lambda}} \mathcal{F}(\psi, \boldsymbol{\lambda}), \\ \boldsymbol{\lambda} &\to \boldsymbol{\lambda} + \boldsymbol{v}, \end{aligned} \tag{S39}$$

where here η is the *learning rate* and γ the *momentum*. The use of the auxiliary parameter $\boldsymbol{v}$ during the training discourages sudden changes of direction, and can make the training significantly more efficient [41].

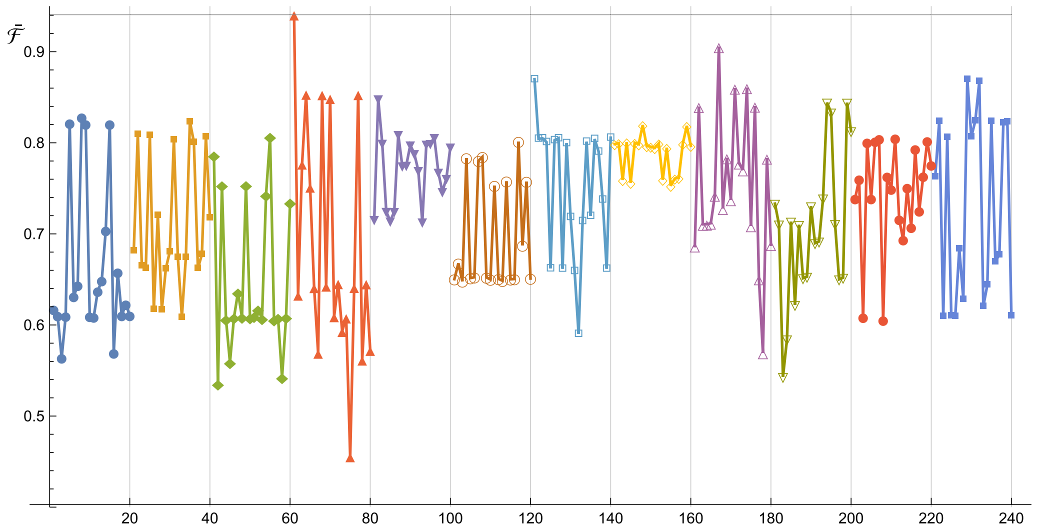

Figure B8. Training results for the Fredkin gate with a generator containing one-qubit interactions and two-qubit interactions of the form $J_{ij}(X_iX_j + Y_iY_j)$. Every point shows the final fidelity obtained at the end of a training procedure. The hyperparameters, as well as the total number of training iterations, are kept the same in all the training instances shown here. The initial parameters' values are the same within each horizontal sector, but changed between different sectors. The initial parameters' values within each sector have been chosen as all equal to c (that is, $\lambda_i = c$ for all i). The values of c are $0, 1, \ldots, 10$, with $c = 0$ in the leftmost sector and $c = 10$ in the last to rightmost one. The rightmost sector contains the results of training attempts with the initial values chosen at random (that is, with λ_i sampled according to the uniform normal distribution, independently for each i). The greatest reached fidelity, obtained with initial parameters' values $\lambda_i = 3$, is $\mathcal{F} \simeq 0.94$.

While the cost function $\mathcal{F}$ is always real, some of the intermediate calculations needed to compute it involve complex numbers. While this poses no fundamental problems, many of the machine learning (ML) libraries do not support AD over functions with complex inputs or outputs. We worked around this problem using a similar trick to the one reported in [68]. In particular, to use the existing framework, we mapped the problem into one involving only real numbers. To do this, we map complex matrices into real ones via the bijection $A \mapsto \mathfrak{Re}(A) \equiv \mathbb{1} \otimes A_R - i\sigma_y \otimes A_I$, where A_R and A_I are the real and imaginary parts of A, respectively. At the same time, state vectors are to be mapped to $\Psi \mapsto \mathfrak{Re}(\Psi) \equiv (\Psi_R, \Psi_I)^T$. It is easy to verify that with this mapping $A\Psi \mapsto \mathfrak{Re}(A\Psi) = \mathfrak{Re}(A)\mathfrak{Re}(\Psi)$, so that all calculations can be equivalently be carried out with the real versions of matrices and vectors.

More specifically, the employed algorithm involves the following steps:

(a) Choose an initial set of parameters $\boldsymbol{\lambda}$ (randomly, or specific values if one has an idea of where a solution might be). A number of other hyperparameters have to be decided at this step, depending on the exact SGD method used. In particular, for mini-batch SGD with momentum and decreasing learning rate, one has to decide the momentum γ, the initial value of η, the rate at which η decreases during the training, and the size N_b of the batches of states used for every gradient descent step.

(b) Repeat the following loop N_e times, or until a satisfactory result is obtained. Each such iteration is conventionally named an *epoch*. Another hyperparameter to be chosen beforehand is the number of training states N_{tr} to be used in each epoch. Once this is fixed, every epoch will involve a number N_{tr}/N_e of gradient descent steps, each one using N_e states for a single gradient calculation. N_e random training states are sampled, to be used during the epoch.
   1. Pick N_b of the N_e training states.
   2. Forward-propagate each state of the sample, and then backpropagate the gradients, thus computing the average gradient over the mini-batch $\nabla_{\boldsymbol{\lambda}}\mathcal{F}(\boldsymbol{\lambda})$.
   3. Update the coupling strengths λ as per equation (S5).
   4. Return to point (1).

B4. Results

A sample of training results for Toffoli, Fredkin, and 'double Fredkin' gates are given in figure 1. In figures B2–B4 are shown the training histories of the parameters for eight different solutions for Toffoli, Fredkin and *double Fredkin*, respectively. These illustrate how quickly the networks converge for different

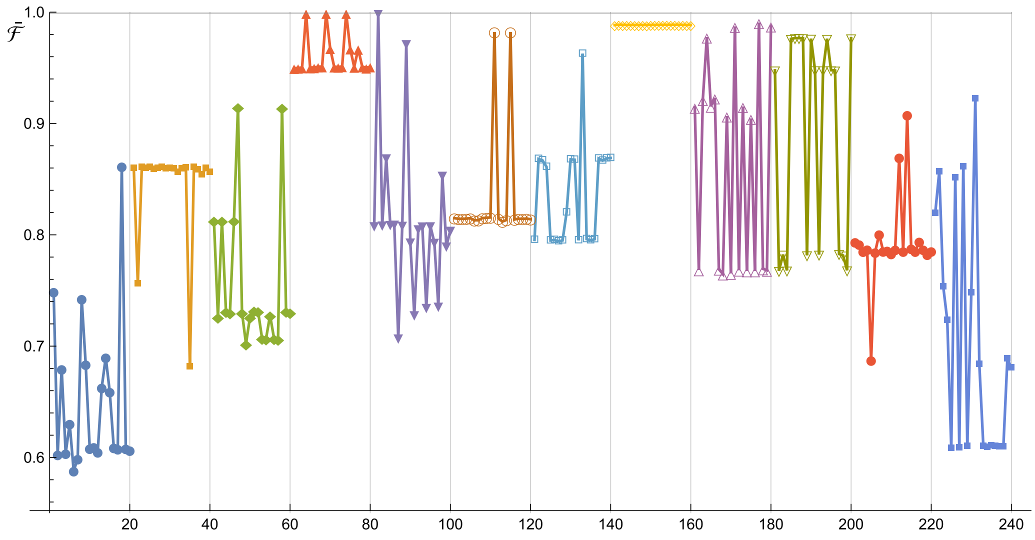

Figure B9. Training results for the Fredkin gate with a generator containing all one-qubit interactions and two-qubit interactions of the form $J_{ij}^{(1)}X_iX_j + J_{ij}^{(2)}Y_iY_j$. The initial conditions are chosen as in figure B8. The maximum fidelities obtained are $\mathcal{F} \simeq 0.999$, obtained in multiple instances $c = 3$ and $c = 4$ sectors.

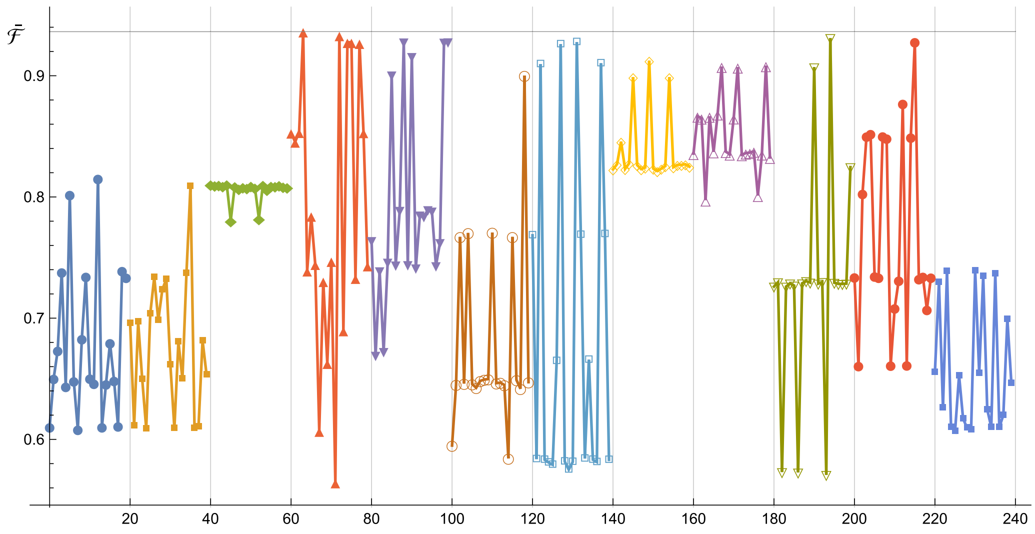

Figure B10. Training results for the Toffoli gate with a generator containing all one-qubit interactions and two-qubit interactions of the form $J_{ij}(X_iX_j + Y_iY_j)$. The initial conditions are chosen as in figure B8. The maximum fidelities obtained are $\mathcal{F} \simeq 0.94$, obtained in the $c = 3$ sector.

initial values of the parameters. In all the shown cases the target gates are obtained with unit fidelity up to numerical precision (that is, all fidelities are between 1×10^{-16} and 1). Different sets of optimization hyperparameters are found to give acceptable solutions. For the trainings shown in this paper we used a dynamically updated learning rate given, for the kth epoch, by $\eta = 1/(1 + k\alpha)$ with the *decay rate* $\alpha = 0.005$. The other hyperparameters were chosen as $\gamma = 0.5$, $N_{\mathrm{b}} = 2$, $N_{\mathrm{tr}} = 200$. Different initial values for the parameters were tested, but in most cases we started the training with either vanishing or random (following a normal distribution) parameters. For the training of the four-qubit gate we found the network to converge sooner to a solution when the parameters were initialised to a positive value (often with all parameters initialised to 4).

In figures B5–B7 we report the behaviour of the fidelity upon changes of the learnt Hamiltonian parameters, for Toffoli, Fredkin and *double Fredkin* gates, respectively. As shown in these plots, the stability of the implemented gates with respect to variations of time and interactions values greatly varies between different solutions, as well as between different parameters in the same solutions.

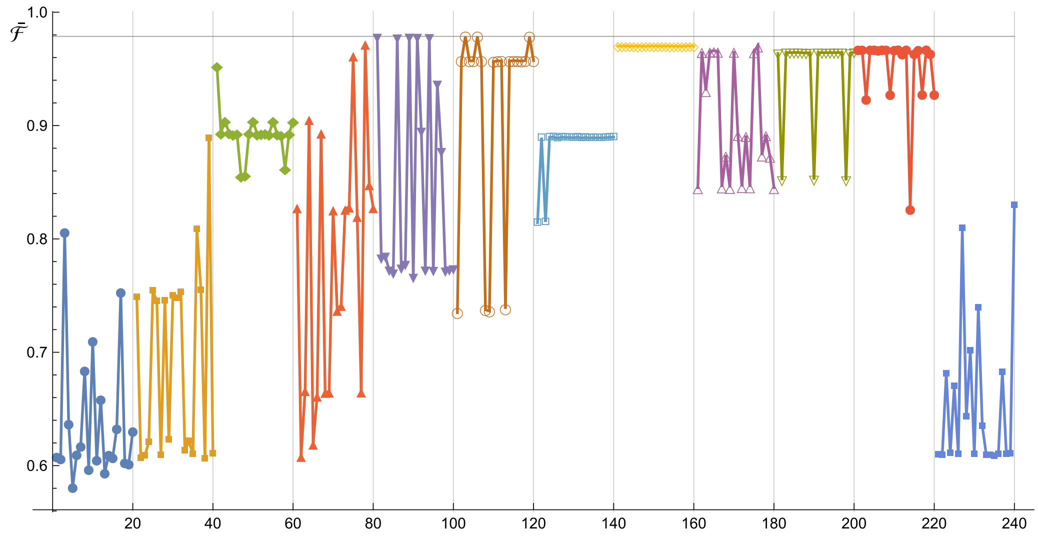

Figure B11. Training results for the Toffoli gate with a generator containing all one-qubit interactions and two-qubit interactions of the form $J_{ij}^{(1)}X_iX_j + J_{ij}^{(2)}Y_iY_j$. The initial conditions are chosen as in figure B8. The maximum fidelities obtained are $\mathcal{F} \simeq 0.98$, obtained in the $c = 4$ and $c = 5$ sectors.

To assess the feasibility of non-trivial gates in more restrictive experimental scenarios, we performed a systematic analysis of the reachability of Fredkin and Toffoli gates when allowing only for single-qubit and $X_iX_j + Y_iY_j$ two-qubit interactions, and in the less restrictive setting of allowing for all X_iX_j and Y_iY_j interactions. The results are shown in figures B8–B11. For the Fredkin gate, in the more restrictive XX interactions setting, the biggest fidelity obtained was $\mathcal{F} \simeq 0.94$, while when allowing for all XX and YY interactions the maximum fidelity obtained was $\mathcal{F} \simeq 0.999$. For the Toffoli gate, the maximum fidelity obtained in the XX scenario was $\mathcal{F} \simeq 0.94$ as well, while when allowing for all XX and YY interactions the best training results corresponded to $\mathcal{F} \simeq 0.98$. To have more consistent results, in all the training instances shown here all the hyperparameters, except for the interaction parameters' initial values, were chosen to have the same value. In particular, each training instance was run for 200 epochs, each one using 200 random quantum states as inputs, divided in batches of two elements. This choice of hyperparameters is mostly empirical, and it is possible for different values to provide better results.

The above provides further evidence for the flexibility of the supervised learning approach, which can produce solutions with good fidelities even in more restrictive scenarios, closer to the capabilities of state of the art experimental architectures. Furthermore, the values of the interaction strengths for many of the presented solutions are found to be compatible with the capabilities of state of the art circuit-QED architectures with gate times of the order of tens of nanoseconds [26, 27]. For instance, the Hamiltonian reported in the supplementary information (https://stacks.iop.org/NJP/22/065001/mmedia) of reference [26], after a suitable adiabatic elimination of the field used to mediate the interaction among various superconducting qubits, returns a model that is very close to the ansatz used in equation (10).

ORCID iDs

Luca Innocenti https://orcid.org/0000-0002-7678-1128
Leonardo Banchi https://orcid.org/0000-0002-6324-8754
Alessandro Ferraro https://orcid.org/0000-0002-7579-6336
Sougato Bose https://orcid.org/0000-0001-8726-0566
Mauro Paternostro https://orcid.org/0000-0001-8870-9134

References

[1] Georgescu I M, Ashhab S and Nori F 2014 Quantum simulation *Rev. Mod. Phys.* **86** 154
[2] Nielsen M A and Chuang I 2002 *Quantum Computation and Quantum Information* (Cambridge, MA: Academic)
[3] Gottesman D 1998 Theory of fault-tolerant quantum computation *Phys. Rev.* A **57** 127
[4] Toffoli T 1980 Reversible Computing *Automata, Languages and Programming* (Lecture Notes in Computer Science vol 85) ed. J. W. de Bakker and J. van Leeuwen (Berlin: Springer) pp 632–44
[5] Shi Y 2002 Both Toffoli and controlled-NOT need little help to do universal quantum computation (arXiv:0205115)

[6] Cory D G, Price M D, Maas W, Knill E, Laflamme R, Zurek W H, Havel T F and Somaroo S S 1998 Experimental quantum error correction *Phys. Rev. Lett.* **81** 2152

[7] Reed M D, DiCarlo L, Nigg S E, Sun L, Frunzio L, Girvin S M and Schoelkopf R J 2012 Realization of three-qubit quantum error correction with superconducting circuits *Nature* **482** 382–5

[8] Monz T, Kim K, Hänsel W, Riebe M, Villar A S, Schindler P, Chwalla M, Hennrich M and Blatt R 2009 Realization of the quantum Toffoli gate with trapped ions *Phys. Rev. Lett.* **102** 040501

[9] Fedorov A, Steffen L, Baur M, da Silva M P and Wallraff A 2011 Implementation of a Toffoli gate with superconducting circuits *Nature* **481** 170–2

[10] Vandersypen L M K, Steffen M, Breyta G, Yannoni C S, Sherwood M H and Chuang I L 2001 Experimental realization of Shor's quantum factoring algorithm using nuclear magnetic resonance *Nature* **414** 883–7

[11] Cubitt T, Montanaro A and Piddock S 2018 Universal quantum Hamiltonians *Proc. Natl Acad. Sci. USA* **115** 9497–9502

[12] d'Alessandro D 2007 *Introduction to Quantum Control and Dynamics* (Boca Raton, FL: CRC Press)

[13] Dong D and Petersen I R 2010 Quantum control theory and applications: a survey *IET Control Theory Appl.* **4** 2651–71

[14] Preskill J 2018 Quantum computing in the NISQ era and beyond *Quantum* **2** 79

[15] Moler C and Van Loan C 2003 Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later *SIAM Rev.* **45** 3–49

[16] Banchi L, Pancotti N and Bose S 2016 Quantum gate learning in qubit networks: Toffoli gate without time-dependent control *npj Quantum Inf.* **216019** 1–6

[17] Al-Mohy A H and Higham N J 2011 Computing the action of the matrix exponential, with an application to exponential integrators *SIAM J. Sci. Comput.* **33** 488–511

[18] Bishop C M 2006 *Pattern Recognition and Machine Learning* (New York: Springer)

[19] Bottou L 1998 Online learning and stochastic approximations *On-line Learn.Neural Netw.* **17** 142

[20] Baydin A G, Pearlmutter B A, Radul A A and Mark Siskind J 2018 Automatic differentiation in machine learning: a survey *J. Mach. Learn. Res.* **18** 1–43 http://jmlr.org/papers/v18/17-468.html

[21] Bartholomew-Biggs M, Brown S, Christianson B and Dixon L 2000 Automatic differentiation of algorithms *J. Comput. Appl. Math.* **124** 171–90

[22] Wengert R E 1964 A simple automatic derivative evaluation program *Commun. ACM* **7** 463–4

[23] Bischof C H, Bücker H M, Paul H, Naumann U and Jean U 2008 *Advances in Automatic Differentiation* (Berlin: Springer)

[24] Man-Hong Y and Bose S 2005 Perfect state transfer, effective gates, and entanglement generation in engineered bosonic and fermionic networks *Phys. Rev.* A **71** 032310

[25] Kay A 2009 A review of perfect state transfer and its application as a constructive tool *Int. J. Quantum Inf.* **8** 641–76

[26] Anton P *et al* 2018 Studying light-harvesting models with superconducting circuits *Nat. Commun.* **9** 904

[27] Krantz P, Kjaergaard M, Yan F, Orlando T P, Gustavsson S and Oliver W D 2019 A quantum engineer's guide to superconducting qubits *Appl. Phys. Rev.* **6** 021318

[28] Friedland S, Nocedal J and Overton M L 1987 The formulation and analysis of numerical methods for inverse eigenvalue problems *SIAM J. Numer. Anal.* **24** 634–67

[29] Zahedinejad E, Ghosh J and Sanders B C 2015 Designing high-fidelity single-shot three-qubit gates: a machine learning approach *Phys. Rev. Appl.* **6** 054005

[30] Zahedinejad E, Ghosh J and Sanders B C 2015 High-fidelity single-shot Toffoli gate via quantum control *Phys. Rev. Lett.* **114** 200502

[31] Stojanović V M, Fedorov A, Wallraff A and Bruder C 2012 Quantum-control approach to realizing a Toffoli gate in circuit qed *Phys. Rev.* B **85** 054504

[32] Antonio B, Randall J, Hensinger W, Morley G W and Bose S 2015 Classical computation by quantum bits (arXiv:1509.03420)

[33] Patel R B, Ho J, Ferreyrol F, Ralph T C and Pryde G J 2016 A quantum Fredkin gate *Sci. Adv.* **2** e1501531

[34] Søe Loft N J, Kristensen L B, Andersen C K and Zinner N T 2018 Quantum spin transistors in superconducting circuits (arXiv:1802.04292)

[35] Nielsen M A 2002 A simple formula for the average gate fidelity of a quantum dynamical operation *Phys. Lett.* A **303** 249–52

[36] Joseph E, Alicki R and Życzkowski K 2005 Scalable noise estimation with random unitary operators *J. Opt. B: Quantum Semiclass. Opt.* **7** S347–52

[37] Ruder S 2016 An overview of gradient descent optimization algorithms (arXiv:1609.04747)

[38] The Theano Development Team 2016 Theano: a Python framework for fast computation of mathematical expressions (arXiv:1605.02688)

[39] Abadi M *et al* 2015 TensorFlow: large-scale machine learning on heterogeneous systems software available from tensorflow.org

[40] Paszke A *et al* 2017 Automatic Differentiation in PyTorch *NIPS 2017 Autodiff Workshop: The Future of Gradient-based Machine Learning Software and Techniques* (Long Beach, CA, December 9, 2017)

[41] Goh G 2017 Why momentum really works *Distill* https://doi.org/10.23915/distill.00006

[42] Bairey E, Arad I and Lindner N H 2019 Learning a local Hamiltonian from local measurements *Phys. Rev. Lett.* **122** 020504

[43] Valenti A, van Nieuwenburg E, Huber S and Greplova E 2019 Hamiltonian learning for quantum error correction *Phys. Rev. Res.* **1** 033092

[44] Teoh Y H, Drygala M, Melko R G and Islam R 2020 Machine learning design of a trapped-ion quantum spin simulator *Quantum Sci. Technol.* **5** 024001

[45] Evans T J, Harper R and Flammia S T 2019 Scalable Bayesian Hamiltonian learning (arXiv:1912.07636)

[46] Kay A 2006 Perfect state transfer: beyond nearest-neighbor couplings *Phys. Rev.* A **73** 032306

[47] Banchi L, Compagno E and Bose S 2015 Perfect wave-packet splitting and reconstruction in a one-dimensional lattice *Phys. Rev.* A **91** 052323

[48] Genest V X, Vinet L and Zhedanov A 2016 Quantum spin chains with fractional revival *Ann. Phys.* **371** 348–67

[49] Mohri M, Rostamizadeh A and Talwalkar A 2012 *Foundations of Machine Learning* (Cambridge, MA: MIT Press)

[50] Hecht-Nielsen R 1989 Theory of the backpropagation neural network *Int. 1989 Joint Conf. on Neural Networks Vol 1* pp 593–605

[51] Simon H 1998 *Neural Networks: A Comprehensive Foundation* (Upper Saddle River, NJ: Prentice Hall)

[52] Amin M H, Andriyash E, Rolfe J, Kulchytskyy B and Melko R 2018 Quantum Boltzmann machine *Phys. Rev.* X **8** 021050

[53] Wang L 2016 Discovering phase transitions with unsupervised learning *Phys. Rev.* B **94** 195105

[54] Hush M R 2017 Machine learning for quantum physics *Science* **355** 580

[55] Carleo G and Troyer M 2017 Solving the quantum many-body problem with artificial neural networks *Science* **355** 602–6
[56] Carrasquilla J and Melko R G 2017 Machine learning phases of matter *Nat. Phys.* **13** 431–4
[57] Torlai G, Mazzola G, Carrasquilla J, Troyer M, Melko R and Carleo G 2018 Many-body quantum state tomography with neural networks *Nat. Phys.* **14** 447
[58] Broecker P, Assaad F F and Simon T 2017 Quantum phase recognition via unsupervised machine learning (arXiv:1707.00663 [quant-ph])
[59] Deng D-L, Li X and Das Sarma S 2017 Quantum entanglement in neural network states *Phys. Rev.* X **7** 021021
[60] Michael Swaddle , Noakes L, Smallbone H, Salter L and Wang J 2017 Generating three-qubit quantum circuits with neural networks *Phys. Lett.* A **381** 3391–5
[61] Krastanov S and Jiang L 2017 Deep neural network probabilistic decoder for stabilizer codes *Sci. Rep.* **7** 11003
[62] Gray J, Banchi L, Bayat A and Bose S 2018 Machine learning assisted many-body entanglement measurement *Phys. Rev. Lett.* **121** 150503
[63] Banchi L, Bayat A, Verrucchi P and Bose S 2011 Nonperturbative entangling gates between distant qubits using uniform cold atom chains *Phys. Rev. Lett.* **106** 140501
[64] Pedersen L H, Martin Møller N and Mølmer K 2007 Fidelity of quantum operations *Phys. Lett.* A **367** 47–51
[65] Magesan E, Blume-Kohout R and Joseph E 2011 Gate fidelity fluctuations and quantum process invariants *Phys. Rev.* A **84** 012309
[66] Johansson J R, Nation P D and Nori F 2012 Qutip: an open-source Python framework for the dynamics of open quantum systems *Comput. Phys. Commun.* **183** 1760–72
[67] Johansson J R, Nation P D and Nori F 2013 Qutip 2: a Python framework for the dynamics of open quantum systems *Comput. Phys. Commun.* **184** 1234–40
[68] Leung N, Mohamed A, Koch J and Schuster D 2017 Speedup for quantum optimal control from automatic differentiation based on graphics processing units *Phys. Rev.* A **95** 042318