Supervised learning of time-independent Hamiltonians for gate design

Supervised learning of time-independent Hamiltonians for gate design

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

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

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 transforming the task to 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.

Let us consider the synthesis of a quantum operation $\mathcal{G}$ (a *gate*) from the underlying dynamics of a quantum system. Unitarity of $\mathcal{G}$ allows for the identification of a Hermitian generator $\mathcal{H}_\mathcal{G}$ such that $\mathcal{G} = e^{i\mathcal{H}_\mathcal{G}}$ (we assume units such that the simulation time t is dimensionless, and rescale it so that the desired gate is successfully achieved at $t = 1$). In general, such generator typically contains highly non-local interactions that 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 sub-set Γ of physical interactions that can be realized relatively easily and inexpensively. The question that we aim at addressing here is thus: *is it possible to synthesise $\mathcal{G}$ from a generator $\mathcal{H}_\mathcal{G}$ comprising operations drawn only from an assigned Γ?*

This question is very relevant in the context of quantum simulation, for instance, where the problem of general *reachability* of a target dynamics, given a set of physical interactions to be used to construct the simulation strategy, is key [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 and with high fidelity. A notable case is the quantum Toffoli gate, a universal reversible logic three-qubit gate [4] that is optimal for quantum error correction [5–8], and is a key component for reversible arithmetic operations such as modular exponentiation [9]. Unfortunately, the natural dynamics generating a Toffoli gate requires non-local three-qubit interactions, which cannot be easily implemented in experimental architectures. Possible ways to overcome the limitation of gate synthesis, simulation, and reachability, typically consists of a suitable use of the additional processing power offered by *larger Hilbert spaces* [10] encompassing ancillary information carriers, and the embedding of quantum control techniques [11].

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 [12].

In this Letter we show that, by exploiting symmetries and degeneracy of the target gate, and making use of the power of machine learning-enhanced quantum information processing, it is indeed possible to identify successful architectures for arbitrary gate synthesis and simulation. More specifically, given a certain set Γ of physical interactions that can be realized inexpensively in a given physical platform, we introduce three conditions that, if met, produce a Hamiltonian $\tilde{\mathcal{H}}$ comprising only operations drawn from Γ and such that $\mathcal{G} = \exp(i\tilde{\mathcal{H}})$.

On one hand, the abovementioned conditions can be used to find exact gate-design strategies. We show relevant instances of such possibility by devising Toffoli and Fredkin gates using only pairwise interactions. On the other hand, the same conditions provide enhanced numerical ansatz for a speedy design of arbitrary N-qubit gates. In particular, we present a supervised-learning optimisation technique to train qubit networks, and demonstrate algorithm-training instances of three-qubit Toffoli and Fredkin gates. 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 automatic differentiation, 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 addresssing quantum communication via perfect state-transfer approaches [13, 14].

General methodology.– We start our analysis by computing the Hamiltonian $\mathcal{H}_\mathcal{G}$ that generates the target gate $\mathcal{G} = e^{i\mathcal{H}_\mathcal{G}}$. Using the spectral decomposition of $\mathcal{G}$, we have $\mathcal{H}_\mathcal{G} = -iU\,\mathrm{Log}(\Lambda)U^\dagger$, where $\mathcal{G} = U\Lambda U^\dagger$, Log denotes the principal branch of the logarithm, and Λ is a 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 $\mathcal{G}$. In general, the generator $\mathcal{H}_\mathcal{G}$ will contain both *physical* interac-

* linnocenti01@qub.ac.uk

tions, that can be realized easily in a given physical setup, and *unphysical* ones, i.e. dynamics that are not naturally achieved in the chosen experimental platform of the problem. 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 Heisenberg coupling strengths and local magnetic fields. In general, $\tilde{\mathcal{H}}_{\mathcal{G}}(\lambda)$ may also model a system where the original register is coupled to auxiliary degrees of freedom, though we will here 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:

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

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

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

Requirements (1b) and (1c) ensure that $\mathcal{G} = \exp(i\tilde{\mathcal{H}}_{\mathcal{G}} - i\mathcal{H}_{\mathcal{G}})\exp(i\mathcal{H}_{\mathcal{G}})$ and $\exp(i\tilde{\mathcal{H}}_{\mathcal{G}} - i\mathcal{H}_{\mathcal{G}}) = \mathbb{1}$, while condition (1a) reiterates the constraints we are imposing on $\tilde{\mathcal{H}}_{\mathcal{G}}$. While condition (1b) may seem excessively restrictive, this turns out to not be the case. To see this, consider the spectral decomposition $\mathcal{G} = \sum_k \lambda_k \sum_j P_{kj}$ of a general gate $\mathcal{G}$. Here, P_{kj} is the j^{th} trace-1 projector in the k^{th} degenerate subsector of the eigenspace [15]. It follows that $\mathcal{H}_{\mathcal{G}}$ and $\tilde{\mathcal{H}}_{\mathcal{G}}$ have the following expressions

$$\mathcal{H}_{\mathcal{G}} = -i\sum_k \mathrm{Log}(\lambda_k) I_k, \quad \tilde{\mathcal{H}}_{\mathcal{G}} = \mathcal{H}_{\mathcal{G}} + 2\pi \sum_{k,j} \nu_{kj} P_{kj}, \tag{2}$$

Here $I_k = \sum_j P_{kj}$, which is not (in general) an identity (or diagonal) matrix, and we have used the general expression for the logarithm $\log\lambda = \mathrm{Log}\,\lambda + 2\pi i\nu$, with $\nu \in \mathbb{Z}$, applied to every term of the spectral decomposition of $\mathcal{G}$. Therefore, for any choice of $\tilde{\mathcal{H}}_{\mathcal{G}}$ we have $[\tilde{\mathcal{H}}_{\mathcal{G}}, \mathcal{H}_{\mathcal{G}}] = 0$.

The set of conditions (1a)-(1c) considerably simplifies the problem of gate simulation, and can be constructively used in several ways. On one hand, they can be analytically solved in at least some situations of physical interest. On the other hand, they can be used to produce an efficient starting point for numerical optimization techniques. The general procedure is to start from a generically parameterized expression for $\tilde{\mathcal{H}}_{\mathcal{G}}(\boldsymbol{\lambda})$ satisfying condition (1a), and then proceed to use (1b) to both significantly reduce the set of possible interactions and impose constraints on the parameters. The problem is then reduced to the enforcing of the constraints on the spectrum of the generator summarised by condition (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 therefore reduces the task of constrained gate design into an inverse eigenvalue problem, a topic well studied in the field of numerical analysis [16]. More generally, we develop a numerical supervised learning technique to avoid having to directly tackle the solution of the non-trivial eigenvalue problem posed by Eq. (1c). It is worth noting that, while $\tilde{\mathcal{H}}_{\mathcal{G}}$ produces the same unitary evolution given by $\mathcal{H}_{\mathcal{G}}$ at time $t = 1$, the dynamics will in general be different at $0 < t < 1$.

Applications: Toffoli and Fredkin gates.– The quantum Toffoli gate $\mathcal{U}_{\text{Toff}}$ is a control-control-NOT that flips the state of the target qubit (qubit 3 in our notation) when the state of the two controls (qubits 1 and 2) is $|1\rangle_1 \otimes |1\rangle_2$, and acts trivially on qubit 3 otherwise. Its realization is an important step towards the construction of quantum computers [8, 17–19]. A time-independent two-body Hamiltonian that simulates $\mathcal{U}_{\text{Toff}}$ with four qubits has been obtained in [20] using a numerical optimization technique, while three qubits have only been found to make approximate and classical Toffoli gates [21]. Here, following the construction in Eq. (1), we find an analytic solution that requires as few as three qubits. Its generator, obtained by taking the principal value of the logarithm of $\mathcal{U}_{\text{Toff}}$, is

$$\mathcal{H}_{\text{Toff}} = \frac{\pi}{8}(1-\sigma_1^z)(1-\sigma_2^z)(1-\sigma_3^x), \tag{3}$$

whose only three-qubit term is $\propto \sigma_1^z\sigma_2^z\sigma_3^x$, where σ_i^α is the α^{th} Pauli matrix acting on qubit i. We now write a general parametrised generator $\tilde{\mathcal{H}}_{\text{Toff}}$ as

$$\tilde{\mathcal{H}}_{\text{Toff}} = h_0\mathbb{1} + \sum h_{i,\alpha}\sigma_i^\alpha + \sum J_{i,j}^{\alpha,\beta}\sigma_i^\alpha\sigma_j^\beta. \tag{4}$$

The above expression, containing 37 parameters, automatically satisfies condition (1a) in that it corresponds to an $\tilde{\mathcal{H}}_{\text{Toff}}$ without three-qubit interactions. Imposing condition (1b) further removes 12 parameters, leaving us with 25 (see Section I in the Supplementary Material for more details). This number is still too high to easily solve the inverse eigenvalue problem embodied by condition (1c). We thus impose some physically plausible assumptions on the coefficients, in order to obtain a generator with a small enough number of parameters for which condition (1c) can be satisfied *and* the resulting equations are simple enough to be solvable. In particular, we impose $J_{12}^{xz} = J_{12}^{zx} = J_{13}^{xx} = J_{23}^{xx} = 0$, $J_{13}^{zx} = J_{23}^{zx} = \pi/8$, $J_{23}^{zz} = -J_{13}^{zz}$ and $h_{1,2}^z = -\pi/8$. 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 $\mathcal{H}'_{\text{Toff}} \equiv \tilde{\mathcal{H}}_{\text{Toff}} - \mathcal{H}_{\text{Toff}}$, which is the operator on which we have to impose Eq. (1c). Note that the above does not uniquely identify the set of assumptions, and different assumptions leading to different classes of solutions are indeed possible. In the Supplementary Materials we present another example of generator, that is obtained via different assumptions. Imposing the above constraints we obtain

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

The problem is now to find values for the coefficients in Eq. (5) such that $\exp(i\mathcal{H}'_{\text{Toff}}) = \mathbb{1}$, 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 a family of solutions parametrized by the 4 integer coefficients $\nu_1, \nu_2, \nu_3, \nu_4$. The full expression is given in the Supplementary Materials.

A simpler family of solutions depending on a single integer parameter is obtained imposing $\nu_1 = \nu_2 = \nu_3 = 0$ and $\nu_4 = \nu$, and reads

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

As can be directly verified, for non-zero integer values of ν, the above satisfies $\exp(i\tilde{\mathcal{H}}_{\text{Toff}}(\nu)) = \mathcal{U}_{\text{Toff}}$. It is interesting to remark that the generator in Eq. (6) can also be deduced directly by using only properties of the Pauli matrices, as shown in the supplementary materials (while this is harder in the case of the generator provided for the Fredkin in Eq. (8)). Thus, a highly non-trivial three-qubit gate such as Toffoli's, which in principle requires three-body interactions as in Eq. (3), can be obtained exactly *without* three-qubit interactions.

On a similar note, it is possible to use the framework provided by Eqs. (1a) and (1c) to find a Hamiltonian that does not contain three-qubit interaction terms, and generates the Fredkin gate at suitable time. The Quantum Fredkin gate $\mathcal{U}_{\text{Fred}}$ is a three qubits 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 [22, 23]. A time-independent two-body Hamiltonian that simulates $\mathcal{U}_{\text{Fred}}$ with 4 qubits has been obtained in [20] using a numerical optimization technique. We find an analytic solution that requires as few as three qubits. Explicitly

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

where the control qubit is the first one. Clearly, the above Hamiltonian contains both two-body and three-body interactions. We now write down the general parameterized expression $\tilde{\mathcal{H}}_{\mathcal{G}}(\boldsymbol{\lambda})$ for a 3-qubit Hamiltonian containing only pairwise *diagonal* interactions, and imposing Eq. (1b) we cut the number of parameters $\boldsymbol{\lambda}$ down to 22. Imposing some physically plausible additional conditions, like the symmetry of second and third qubit, we finally manage to reduce the number of parameters enough to solve the eigenvalue problem, finding the following solution

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

This proves that also the Fredkin gate, a non-trivial gate with several applications, 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 due to the symmetries of the gate. *De facto*, such degeneracy makes the time-evolution operator 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 heavy 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. Nonetheless, although $\exp(it\tilde{\mathcal{H}}_{\text{Fred}})$ is non-symmetric for most of the evolution times, by construction all the symmetries are restored at $t = 1$, as well as at any subsequent integer times. This example nicely shows that breaking the symmetries of a gate $\mathcal{G}$, exploiting its degenerate space, can help the gate simulation when restricting the set of viable interactions.

Perfect state transfer.– A one-dimensional quantum walk is described by the Hamiltonian $\mathcal{H}_W = \sum_{k=1}^{N-1} J_k |k\rangle\langle k-1| + B_k |k\rangle\langle k|$, 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 k-th site. A quantum walk Hamiltonian $\mathcal{H}_W$ 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 $e^{-it\mathcal{H}_W} = \Xi$, where $\Xi_{kj} = \delta_{k,N-j+1}$ is the reflection matrix. Necessary and sufficient conditions for PST are well understood [13, 14]: firstly, $\mathcal{H}_W$ has to be "mirror-symmetric", that is $[\mathcal{H}_W, \Xi] = 0$, and secondly the eigenvalues $\{E_k\}$ of $\mathcal{H}_W$ should to satisfy the condition $e^{iE_k t} = (\pm 1)^k$.

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 Eq. (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 $\mathcal{H}_W$ as $\mathcal{H}^{\text{phys}}(\lambda)$, where $\lambda = \{J_k, B_k\}$, and defining $\mathcal{H}' = \mathcal{H}_W - \mathcal{H}_\Xi$ following condition (1c), where the physical interactions in $\mathcal{H}_\Xi$ have been reabsorbed into the definition of $\mathcal{H}_W$, one finds that: (i) condition (1b) is equivalent to the mirror-symmetry request $[\mathcal{H}_W, \Xi] = 0$; (ii) from conditions (1c) and from the definitions of $\mathcal{H}_W$ and $\mathcal{H}_\Xi$, one finds that the spectrum of $\mathcal{H}_W$ satisfies $e^{iE_k t} = (\pm 1)^k$ — indeed the eigenvalues of Ξ are $\{0, \pi\}$, so $E_k = \pi(2n_k + 1)$ for integer n_k. It is straigthforward to extend the above proof for more general transfers, such as with long-range interactions [24], or for perfect fractional revivals [25, 26].

Supervised learning approach– We now describe a different methodology to solve the difficult part of the conditions given in Eq. (1), that is imposing the condition on the eigenvalues. 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 much 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. A first exploration of such a method has been laid down in Ref. [20]. However, we have been able to significantly improve on this work, by borrowing from machine learning techiques commonly used to train neural networks. As we show below, this results in a more efficient training and allows for the exploration of a richer set of scenarios.

The problem we set up to solve is a generalized version of the one presented above. Given a target gate $\mathcal{G}$ and a parameterized 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 want to find the set $\boldsymbol{\lambda}_0$ such that $\exp(i\mathcal{H}(\boldsymbol{\lambda}_0)) = \mathcal{G}$. This can be reframed as an optimization problem by considering the fidelity function $\mathcal{F}(\boldsymbol{\lambda}, \psi) \equiv \langle\psi|\mathcal{G}^\dagger \exp(i\mathcal{H}(\boldsymbol{\lambda}))|\psi\rangle$, for an arbitrary state $|\psi\rangle$. Clearly, $\mathcal{F}(\boldsymbol{\lambda}_0, \psi) = 1$ for all $|\psi\rangle$ if and only if $\exp(i\mathcal{H}(\boldsymbol{\lambda}_0)) = \mathcal{G}$. 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. Given that explicit formulas are known for $\bar{\mathcal{F}}$ [27–29], standard optimisation methods can be applied directly to $\bar{\mathcal{F}}$ to find solutions to the gate design problem [1]. This method, however, turns out to be inefficient for the problem at hand, due to the complexity of the underlying parameter space. We therefore turn to a different technique, exploiting how the fidelity landscape changes when changing the state $|\psi\rangle$ [20]. We can indeed 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. By employing a gradient descent technique [31, 32] and different $|\psi\rangle$ at different steps, we implement the following iterative procedure:

1. Choose an initial set of parameters $\boldsymbol{\lambda}$.

2. Generate a random set of input states $|\psi_k\rangle$, $k = 1, ..., N_b$, with N_b the size of the mini-batches chosen beforehand.

3. For each k, compute $\boldsymbol{\nabla}_{\boldsymbol{\lambda}}\mathcal{F}(\boldsymbol{\lambda}, \psi_k)$. The use of machine learning frameworks like Theano [33], TensorFlow [34], or PyTorch [35] (among others), enabled the calculation of gradients automatically from the chain rule, thus avoiding numerical errors arising from numerical differentiation algorithms.

4. Update the coupling strengths $\boldsymbol{\lambda}$. We here do this using the so-called momentum gradient descent method [36], corresponding to the following updating rule:

$$\begin{aligned} \boldsymbol{v} &\to \gamma \boldsymbol{v} + \eta \boldsymbol{\nabla}_{\boldsymbol{\lambda}}\mathcal{F}(\boldsymbol{\lambda}, \psi_k), \\ \boldsymbol{\lambda} &\to \boldsymbol{\lambda} + \boldsymbol{v}, \end{aligned} \tag{9}$$

   where the *learning rate* η and the *momentum term* γ are hyperparameters to be chosen beforehand. The value of the learning rate is also chosen to be decreasing with the iteration number.

5. Return to point 2, until a satisfactory value of the fidelity is obtained.

A more in-depth explanation of the techniques involved is provided in the Supplementary Materials. To find the interaction parameters implementing a Toffoli gate, using only one-qubit interactions and two-qubit diagonal interactions (that is, interactions of the form $\sigma_1^\alpha \sigma_2^\alpha$), we start the numerical training from the Hamiltonian obtained by imposing Eq. (1b) on the parametrized Hamiltonian containing the required interactions, which has the form

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

Starting the training from Eq. (10), many different solutions can be found, depending on the chosen initial conditions and the random states that are used at each run. In Fig. 1 **(a)**, several different solutions are shown, proving that it is indeed possible to implement a Toffoli gate using only pairwise diagonal interactions. This analysis can be extended to the case of 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 Fig. 1 **(b)**.

More generally, we need not limit ourselves to the training of three-qubit networks. To illustrate this, we provide yet another example of successful application of our framework, this time to implement a non-trivial unitary evolution over *four* qubits. In particular, we successfully train a four-qubit network to implement the *doubly-controlled* Fredkin gate $\mathcal{U}_{FF}$, defined as $\mathcal{U}_{FF} \equiv |0\rangle\langle 0| \otimes \mathcal{U}_{\text{Fred}} + |1\rangle\langle 1| \otimes \mathcal{U}_{\overline{\text{Fred}}}$, where $\mathcal{U}_{\overline{\text{Fred}}}$ denotes a Fredkin gate in which the control qubit is the third one, and the target ones the first two. It turns out that this four-qubit gate can be implemented using no more than two-qubit interactions, and that furthermore this set can be restricted to only consider *diagonal* ones. Some examples of such solutions are shown in Fig. 1 **(c)**. Note that to obtain these results no ad-hoc reasoning was used, nor ansatz such as those used to derive Eqs. (6) and (8) were needed. This, in particular, makes it easy to test any hypothesis such as "can gate X be implemented using only set of interaction Y" without having to go through extra ad-hoc calculations.

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 super conductive qubits.

Acknowledgements– 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 DfE-SFI Investigator Programme (grant 15/IA/2864), the H2020 Collaborative

[1] Note that in principle a more reliable figure of merit to evaluate the performance of a gate would be the diamond distance [30]. However, in all the instances we considered, the two quantities are actually equivalent since we always obtain unit fidelity (within numerical errors).

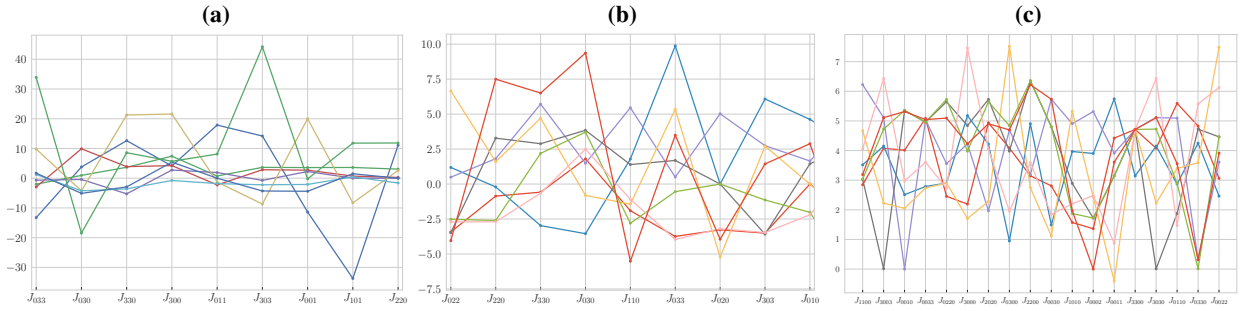

FIG. 1. Eight different sets of interaction parameters generating the Toffoli gate [panel **(a)**] and the Fredkin one [panel **(b)**]. For each shown solution the training was started from the ansatz provided by Eq. (10), and the analogous equation for the Fredkin, respectively. Panel **(c)**: Eight 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 the Supplementary Materials for the details of the optimisations.

Project TEQ (grant 766900), and the Royal Society. LI is partially supported by Fondazione Angelo della Riccia.

[1] I. M. Georgescu, S. Ashhab, and F. Nori, "Quantum simulation," Rev. Mod. Phys. **86**, 154 (2014).

[2] Michael A Nielsen and Isaac Chuang, *Quantum computation and quantum information* (Cambridge Academic Press, 2002).

[3] Daniel Gottesman, "Theory of fault-tolerant quantum computation," Physical Review A **57**, 127 (1998).

[4] Yaoyun Shi, "Both toffoli and controlled-not need little help to do universal quantum computation," arXiv:0205115 (2002).

[5] D. G. Cory, M. D. Price, W. Maas, E. Knill, R. Laflamme, W. H. Zurek, T. F. Havel, and S. S. Somaroo, "Experimental quantum error correction," Physical Review Letters **81**, 2152 (1998).

[6] M. D. Reed, L. DiCarlo, S. E. Nigg, L. Sun, L. Frunzio, S. M. Girvin, and R. J. Schoelkopf, "Realization of three-qubit quantum error correction with superconducting circuits," Nature **482**, 382–385 (2012).

[7] T. Monz, K. Kim, W. Hänsel, M. Riebe, A. S. Villar, P. Schindler, M. Chwalla, M. Hennrich, and R. Blatt, "Realization of the quantum toffoli gate with trapped ions," Physical Review Letters **102**, 040501 (2009).

[8] A. Fedorov, L. Steffen, M. Baur, M. P. da Silva, and A. Wallraff, "Implementation of a toffoli gate with superconducting circuits," Nature **481**, 170–172 (2011).

[9] Lieven M. K. Vandersypen, Matthias Steffen, Gregory Breyta, Costantino S. Yannoni, Mark H. Sherwood, and Isaac L. Chuang, "Experimental realization of shor's quantum factoring algorithm using nuclear magnetic resonance," Nature **414**, 883–887 (2001).

[10] T. Cubitt, A. Montanaro, and S. Piddock, "Universal quantum hamiltonians," arXiv:1701.05182 (2017).

[11] Domenico d'Alessandro, *Introduction to quantum control and dynamics* (CRC press, 2007).

[12] J. Preskill, "Quantum computing in the nisq era and beyond," ArXiv e-prints (2018), arXiv:1801.00862 [quant-ph].

[13] Man-Hong Yung and Sougato Bose, "Perfect state transfer, effective gates, and entanglement generation in engineered bosonic and fermionic networks," Physical Review A **71**, 032310 (2005).

[14] Alastair Kay, "A review of perfect state transfer and its application as a constructive tool," International Journal of Quantum Information **08**, 641–676 (2009).

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

[16] S Friedland, J Nocedal, and M L Overton, "The formulation and analysis of numerical methods for inverse eigenvalue problems," SIAM Journal on Numerical Analysis **24**, 634–667 (1987).

[17] Ehsan Zahedinejad, Joydip Ghosh, and Barry C Sanders, "Designing high-fidelity single-shot three-qubit gates: A machine learning approach," Physical Review Applied **6** (2015).

[18] Ehsan Zahedinejad, Joydip Ghosh, and Barry C. Sanders, "High-fidelity single-shot toffoli gate via quantum control," Physical Review Letters **114**, 200502 (2015).

[19] Vladimir M Stojanović, A Fedorov, A Wallraff, and C Bruder, "Quantum-control approach to realizing a toffoli gate in circuit qed," Physical Review B **85**, 054504 (2012).

[20] Leonardo Banchi, Nicola Pancotti, and Sougato Bose, "Quantum gate learning in qubit networks: Toffoli gate without time-dependent control," NPJ Quantum Information **2**, 16019 (2016).

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

[22] Raj B. Patel, Joseph Ho, Franck Ferreyrol, Timothy C. Ralph, and Geoff J. Pryde, "A quantum fredkin gate," Science Advances **2**, e1501531 (2016).

[23] Niels Jakob Søe Loft, Lasse Bjørn Kristensen, Christian Kraglund Andersen, and Nikolaj T Zinner, "Quantum spin transistors in superconducting circuits," arXiv:1802.04292 (2018).

[24] Alastair Kay, "Perfect state transfer: Beyond nearest-neighbor couplings," Physical Review A **73**, 032306 (2006).

[25] Leonardo Banchi, Enrico Compagno, and Sougato Bose, "Perfect wave-packet splitting and reconstruction in a one-dimensional lattice," Physical Review A **91**, 052323 (2015).

[26] Vincent X. Genest, Luc Vinet, and Alexei Zhedanov, "Quantum spin chains with fractional revival," Annals of Physics **371**, 348–367 (2016).

[27] Leonardo Banchi, Abolfazl Bayat, Paola Verrucchi, and Sougato Bose, "Nonperturbative entangling gates between distant qubits using uniform cold atom chains," Physical Review Letters **106**, 140501 (2011).

[28] Line Hjortshøj Pedersen, Niels Martin Møller, and Klaus Mølmer, "Fidelity of quantum operations," Physics Letters A **367**, 47–51 (2007), quant-ph/0701138.

[29] Easwar Magesan, Robin Blume-Kohout, and Joseph Emerson, "Gate fidelity fluctuations and quantum process invariants," Physical Review A **84** (2011), 10.1103/physreva.84.012309.

[30] A Yu Kitaev, "Quantum computations: algorithms and error correction," Russian Mathematical Surveys **52**, 1191–1249 (1997).

[31] Christopher M Bishop, *Pattern recognition and machine learning* (Springer-Verlag New York, 2006).

[32] Sebastian Ruder, "An overview of gradient descent optimization algorithms," arXiv:1609.04747 (2016).

[33] The Theano Development Team, "Theano: A python framework for fast computation of mathematical expressions," (2016), arXiv:1605.02688 [cs.SC].

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

[35] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer, "Automatic differentiation in pytorch," (2017).

[36] Gabriel Goh, "Why momentum really works." Distill **2** (2017), 10.23915/distill.00006.

[37] Atilim Gunes Baydin, Barak A Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind, "Automatic differentiation in machine learning: a survey," (2015).

[38] Michael Bartholomew-Biggs, Steven Brown, Bruce Christianson, and Laurence Dixon, "Automatic differentiation of algorithms," Journal of Computational and Applied Mathematics **124**, 171–190 (2000).

[39] R. E. Wengert, "A simple automatic derivative evaluation program," Communications of the ACM **7**, 463–464 (1964).

[40] Christian H Bischof, H Martin Bücker, Paul Hovland, Uwe Naumann, and Jean Utke, *Advances in automatic differentiation* (Springer, 2008).

[41] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar, *Foundations of machine learning* (MIT press, 2012).

[42] R. Hecht-Nielsen, "Theory of the backpropagation neural network," International 1989 Joint Conference on Neural Networks , 593–605 vol.1 (1989).

[43] Simon Haykin, *Neural Networks: A Comprehensive Foundation*, edited by Pearson (1994).

[44] Mohammad H Amin, Evgeny Andriyash, Jason Rolfe, Bohdan Kulchytskyy, and Roger Melko, "Quantum boltzmann machine," (2016).

[45] Lei Wang, "Discovering phase transitions with unsupervised learning," Physical Review B **94** (2016), 10.1103/physrevb.94.195105.

[46] Michael R. Hush, "Machine learning for quantum physics," Science **355**, 580–580 (2017).

[47] Giuseppe Carleo and Matthias Troyer, "Solving the quantum many-body problem with artificial neural networks," Science **355**, 602–606 (2017).

[48] Juan Carrasquilla and Roger G. Melko, "Machine learning phases of matter," Nature Physics **13**, 431–434 (2017).

[49] Giacomo Torlai, Guglielmo Mazzola, Juan Carrasquilla, Matthias Troyer, Roger Melko, and Giuseppe Carleo, "Many-body quantum state tomography with neural networks," (2017).

[50] Peter Broecker, Fakher F Assaad, and Simon Trebst, "Quantum phase recognition via unsupervised machine learning," (2017), arXiv:1707.00663 [quant-ph].

[51] Dong-Ling Deng, Xiaopeng Li, and S. Das Sarma, "Quantum entanglement in neural network states," Physical Review X **7**, 021021 (2017).

[52] Michael Swaddle, Lyle Noakes, Harry Smallbone, Liam Salter, and Jingbo Wang, "Generating three-qubit quantum circuits with neural networks," Physics Letters A **381**, 3391–3395 (2017), arXiv:1703.10743 [quant-ph].

[53] Stefan Krastanov and Liang Jiang, "Deep neural network probabilistic decoder for stabilizer codes," Scientific reports **7**, 11003 (2017), arXiv:1705.09334 [quant-ph].

[54] Johnnie Gray, Leonardo Banchi, Abolfazl Bayat, and Sougato Bose, "Measuring entanglement negativity," (2017), arXiv:1709.04923 [quant-ph].

[55] Gabriel Goh, "Why momentum really works." Distill **2** (2017), 10.23915/distill.00006.

[56] Sebastian Ruder, "An overview of gradient descent optimization algorithms," (2016), arXiv:1609.04747 [cs.LG].

[57] J.R. Johansson, P.D. Nation, and Franco Nori, "Qutip: An open-source python framework for the dynamics of open quantum systems," Computer Physics Communications **183**, 1760–1772 (2012).

[58] J.R. Johansson, P.D. Nation, and Franco Nori, "Qutip 2: A python framework for the dynamics of open quantum systems," Computer Physics Communications **184**, 1234–1240 (2013).

[59] Nelson Leung, Mohamed Abdelhafez, Jens Koch, and David Schuster, "Speedup for quantum optimal control from automatic differentiation based on graphics processing units," Physical Review A **95** (2017), 10.1103/physreva.95.042318.

Supplementary Material

In Section I of this Supplementary Materials we explore further the solution framework introduced in the main text. All presented results are readily reproducible via the Mathematica code freely accessible in the GitHub repository at lucainnocenti/quantum-gate-learning-1803.07119-Mathematica-code. In Section II are given the details of the supervised learning approach to the gate design problem. Again, all results are reproducible using the code available in the GitHub repository at lucainnocenti/quantum-gate-learning-1803.07119.

I. ANALYTICAL APPROACH

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

In Section I A 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 Eq. (1c) can be used to *rule out* that a gate can be implemented with a specific set of interactions. In Section I B we show how to use our framework to derive Hamiltonians implementing the Toffoli gate, using only one- and two-qubit interactions. In Section I C we show how one of the solutions obtained in Section I B could have been obtained via direct analytical reasoning, without any knowledge of the conditions given in the main text. 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 Eq. (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.

A. Proof that CNOT needs 2-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

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

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 following basis set of trace-1 projectors for the degenerate space: $\{Z_1^+Z_2^+, Z_1^+Z_2^-, Z_1^-X_2^+\}$, whereas the fourth projector is bound to be $Z_1^-X_2^-$. The corresponding principal Hamiltonian $\mathcal{H}_{\text{CNOT}}$, obtained by directly taking the logarithm of Eq. (S1), is:

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

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:

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

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 i-th projector. Note how the Z_1X_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 $R = \sum_{i,j=1}^3 r_{ij} \left|+1_i\right\rangle\!\left\langle+1_j\right|$, with $|+1_i\rangle$ the i-th eigenvector in a fixed base of the degenerate space. The problem is then that of finding a unitary R and integers ν_i such that

$$\begin{aligned}\nu_1 R(Z_1^+Z_2^+)R^\dagger + \nu_2 R(Z_1^+Z_2^-)R^\dagger + \nu_3 R(Z_1^-X_2^+)R^\dagger \\ +(\nu_4 + 1/2)Z_1^-X_2^-\end{aligned}$$

does not contain 2-qubit interactions. The solution of this problem is non-trivial, mostly due to the many (9 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 $\mathcal{H}_{\text{CNOT}}$ and contains only 1-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 shown in the main text, for $\tilde{\mathcal{H}}$ to correctly generate CNOT, it must commute with the principal generator $\mathcal{H}_{\text{CNOT}}$. 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 $\mathcal{H}_{\text{CNOT}} - \tilde{\mathcal{H}}$ 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.

B. Toffoli gate: derivation through conditions

We show here the details of how, using Eqs. (1), (2) and (3) in the main text, we can obtain a family of Hamiltonian generators for the Toffoli gate, containing only single- and two-qubit interactions.

The Toffoli gate can be written as

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

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

$$\mathcal{H}_{\text{Toff}} = (1\text{- and 2-qubit terms}) - \pi/8\, Z_1Z_2X_3. \tag{S8}$$

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

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

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

$$\begin{aligned}\tilde{\mathcal{H}}_{\text{Toff}} =& h_0\mathbb{1} + h_3^xX_3 + h_1^zZ_1 + h_2^zZ_2 + \\ & J_{13}^{zx}Z_1X_3 + J_{23}^{zx}Z_2X_3 + J_{12}^{zz}Z_1Z_2 + \\ & (J_{13}^{xx}X_1 + J_{23}^{xx}X_2)(1+X_3) + \\ & (J_{12}^{zx}X_2 + J_{13}^{zz}Z_3)(1+Z_1) + \\ & (J_{12}^{xz}X_1 + J_{23}^{zz}Z_3)(1+Z_2).\end{aligned} \tag{S10}$$

As can be directly verified, the above satisfies $[\tilde{\mathcal{H}}_{\text{Toff}}, \mathcal{H}_{\text{Toff}}] = 0$ 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 $\tilde{\mathcal{H}}_{\text{Toff}} - \mathcal{H}_{\text{Toff}}$ all integer multiples of $2\pi i$, but the associated calculations are made hard by the many parameters involved. It turns out however that we can make a number of further assumptions on the form of $\tilde{\mathcal{H}}_{\text{Toff}}$, and still obtain a viable family of solutions. One such set of assumptions, that leads 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 Eq. (S10) becomes

$$\begin{aligned}\tilde{\mathcal{H}}_{\text{Toff}} =& h_0\mathbb{1} + h_3^xX_3 - \pi/8(Z_1+Z_2)(1-X_3) + \\ & J_{12}^{zz}Z_1Z_2 + J_{13}^{xx}(X_1+X_2)(1+X_3) + \\ & J_{13}^{zz}(Z_1-Z_2)Z_3.\end{aligned} \tag{S11}$$

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

$$\begin{aligned}\mathcal{H}'_{\text{Toff}} =& \pi/8\, Z_1Z_2X_3 + (h_0 - \pi/8)\mathbb{1} + \\ & (h_3^x + \pi/8)X_3 + (J_{12}^{zz} - \pi/8)Z_1Z_2 + \\ & J_{13}^{zz}(Z_1 - Z_2)Z_3.\end{aligned} \tag{S12}$$

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

$$\begin{aligned}\tilde{\mathcal{H}}_{\text{Toff}} =& \frac{\pi}{8}\Big[1 + 4\left(\nu_1 + \nu_2 + 2\nu_3 + \sqrt{(\nu_3 - \nu_4)^2}\right) \\ & - (Z_1 + Z_2)(1 - X_3) + X_3(-2 - 8\nu_1 + 8\nu_2) \\ & + 4Z_1Z_2\left(1/4 + \nu_1 + \nu_2 - 2\nu_3 - \sqrt{(\nu_3 - \nu_4)^2}\right) \\ & + (Z_2 - Z_1)Z_3\ \sqrt{c(\nu_1,\nu_2,\nu_3,\nu_4)}\Big],\end{aligned} \tag{S13}$$

with

$$\begin{aligned}c(\nu_1,\nu_2,\nu_3,\nu_4) =& -(1 + 4\nu_1 - 4\nu_2 + 4\nu_3 - 4\nu_4) \\ & \times (1 + 4\nu_1 - 4\nu_2 - 4\nu_3 + 4\nu_4) = \\ =& -[(1 + 4\nu_{12})^2 - (4\nu_{34})^2],\end{aligned} \tag{S14}$$

for all integer values of ν_i such that $c(\nu_1,\nu_2,\nu_3,\nu_4) \geq 0$. The corresponding spectrum of $\mathcal{H}'_{\text{Toff}} = \tilde{\mathcal{H}}_{\text{Toff}} - \mathcal{H}_{\text{Toff}}$ is

$$\begin{aligned}\lambda_1 &= \lambda_2 = 2\pi\nu_1, \\ \lambda_3 &= \lambda_4 = 2\pi\nu_2, \\ \lambda_5 &= \lambda_6 = 2\pi\nu_3, \\ \lambda_7 &= \lambda_8 = 2\pi(\nu_3 + |\nu_3 - \nu_4|),\end{aligned} \tag{S15}$$

while the spectrum of $\tilde{\mathcal{H}}_{\text{Toff}}$ 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 $\mathcal{H}_{\text{Toff}}$, while all the other eigenvalues correspond to eigenvectors orthogonal to this one. More specifically, we have

$$\begin{aligned}|\lambda_1\rangle &= |0,0,-\rangle, \\ |\lambda_2\rangle &= |1,1,-\rangle, \\ |\lambda_3\rangle &= |1,1,+\rangle, \\ |\lambda_4\rangle &= |0,0,+\rangle, \\ |\lambda_5\rangle &= |1,0\rangle \otimes N_5\left[(a-b)\,|0\rangle + |1\rangle\right], \\ |\lambda_6\rangle &= |0,1\rangle \otimes N_6\left[(a+b)\,|0\rangle + |1\rangle\right], \\ |\lambda_7\rangle &= |1,0\rangle \otimes N_6\left[(a+b)\,|0\rangle - |1\rangle\right], \\ |\lambda_8\rangle &= |0,1\rangle \otimes N_5\left[(a-b)\,|0\rangle - |1\rangle\right],\end{aligned} \tag{S16}$$

where

$$a = \frac{|\bar{\nu}_{34}|}{1 + \bar{\nu}_{12}}, \qquad b = \frac{\sqrt{\bar{\nu}_{34}^2 - (\bar{\nu}_{12} + 1)^2}}{1 + \bar{\nu}_{12}}, \tag{S17}$$

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 $c(\nu_1,\nu_2,\nu_3,\nu_4) \geq 0$ cannot be satisfied unless $\nu_3 \neq \nu_4$. This in turn, looking at

Eq. (S16), 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 3-qubit terms, we find

$$\begin{aligned} P_7 &\simeq -N_6^2 \frac{Z_1Z_2}{4}\left[((a+b)^2-1)\frac{Z_3}{2} - (a+b)X_3\right], \\ P_8 &\simeq -N_5^2 \frac{Z_1Z_2}{4}\left[((a-b)^2-1)\frac{Z_3}{2} - (a-b)X_3\right]. \end{aligned} \tag{S18}$$

The term in the Hamiltonian to which these two projectors contribute is $2\pi\nu_{3,4}(P_7+P_8)$, with $\nu_{3,4} = \nu_3 + |\nu_3 - \nu_4|$. A little algebra reveals that the 3-qubit terms in P_7+P_8 are

$$\begin{aligned} -\frac{Z_1Z_2Z_3}{8}\left[N_6^2((a+b)^2-1) + N_5^2((a-b)^2-1)\right] \\ +\frac{Z_1Z_2X_3}{4}\left[N_6^2(a+b) + N_5^2(a-b)\right]. \end{aligned} \tag{S19}$$

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

$$P_7 + P_8 = (...) + Z_1Z_2X_3\frac{1+4(\nu_1-\nu_2)}{16|\nu_3-\nu_4|}. \tag{S20}$$

Substitution of the appropriate values of ν_i shows that the above term can be used to generate the 3-qubit factor $\pi/8\ Z_1Z_2X_3$, *without introducing additional 3-qubit factors*. In Box 1 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 2-qubit interactions for the CCY gate (that is, the gate that applies Y to the third qubit conditionally to the first 2 qubits being in the $|1\rangle$ state).

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

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

where

$$\begin{aligned} \sigma_{01} &\equiv \frac{(1+4\nu_{12})X + \sqrt{c}Z}{4|\nu_{34}|}, \\ \sigma_{10} &\equiv \frac{(1+4\nu_{12})X - \sqrt{c}Z}{4|\nu_{34}|}. \end{aligned} \tag{S21}$$

It can be verified that for all values of $\nu_1, \nu_2, \nu_3, \nu_4$, 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

$$\begin{aligned} \frac{\pi}{8}\Big[1 + 4|\nu_4| - 2X_3 - Z_1 - Z_2 + (Z_1+Z_2)X_3 \\ +Z_1Z_2(1-4|\nu_4|) + (Z_2-Z_1)Z_3\sqrt{16\nu_4^2-1}\Big], \end{aligned} \tag{S22}$$

for all $\nu_4 \neq 0$. It is interesting to look at the explicit form of the exponentials generated by this class generators. Computing $\exp\left(i\tilde{\mathcal{H}}t\right)$ with $\tilde{\mathcal{H}}$ as in Eq. (S22), we get the following unitary:

$$\begin{pmatrix} \mathbb{1}_2 & \mathbb{0} & \mathbb{0} & \mathbb{0} \\ \mathbb{0} & S(t,\nu_4) & \mathbb{0} & \mathbb{0} \\ \mathbb{0} & \mathbb{0} & S(t,\nu_4) & \mathbb{0} \\ \mathbb{0} & \mathbb{0} & \mathbb{0} & X(t) \end{pmatrix}, \tag{S23}$$

where

$$S(t,\nu_4) = \begin{pmatrix} a+b & c \\ c & a-b \end{pmatrix}, \tag{S24}$$

$$\begin{aligned} a &= \frac{1+e^{2i\pi t\nu_4}}{2}, \qquad c = \frac{1-e^{2i\pi t\nu_4}}{8\nu_4}, \\ b &= \frac{(-1+e^{2i\pi t\nu_4})\sqrt{16\nu_4^2-1}}{8\nu_4}, \end{aligned} \tag{S25}$$

and

$$X(t) = \frac{1}{2}\begin{pmatrix} (1+e^{i\pi t}) & (1-e^{i\pi t}) \\ (1-e^{i\pi t}) & (1+e^{i\pi t}) \end{pmatrix} \tag{S26}$$

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

$$\begin{pmatrix} \mathbb{1} & & & & \\ & e^{2i\pi t\nu_4} & & & \\ & & 1 & & \\ & & & e^{2i\pi t\nu_4} & \\ & & & & 1 \\ & & & & & X(t) \end{pmatrix}, \tag{S27}$$

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

$$\exp(it\mathcal{H}_{\text{Toff}}) = \begin{pmatrix} \mathbb{1} & & & \\ & \mathbb{1} & & \\ & & \mathbb{1} & \\ & & & X(t) \end{pmatrix}. \tag{S28}$$

A different solution derived from Eq. (S13) is

$$\begin{aligned} \tilde{\mathcal{H}}_{\text{Toff}} = \frac{9\pi}{8} + \frac{3\pi}{4}X_3 - \frac{\pi}{8}(Z_1+Z_2) + \frac{\pi}{8}Z_1Z_2 \\ +\frac{\pi}{8}(Z_1+Z_2)X_3 - \frac{\sqrt{7}\pi}{8}(Z_1-Z_2)Z_3. \end{aligned} \tag{S29}$$

Box 1: Toffoli

$$P_1 = Z_1^+ Z_2^+ X_3^-, \qquad P_2 = Z_1^- Z_2^- X_3^-, \qquad P_3 = Z_1^+ Z_2^+ X_3^+, \qquad P_4 = Z_1^- Z_2^- X_3^+,$$

$$\begin{aligned}
P_5 &= Z_1^- Z_2^+ \frac{1}{2|\bar{\nu}_{34}|}\left[|\bar{\nu}_{34}| + (1+\bar{\nu}_{12})X_3 - \sqrt{-(1+\bar{\nu}_{12})^2 + \bar{\nu}_{34}^2}\, Z_3\right],\\
P_6 &= Z_1^+ Z_2^- \frac{1}{2|\bar{\nu}_{34}|}\left[|\bar{\nu}_{34}| + (1+\bar{\nu}_{12})X_3 + \sqrt{-(1+\bar{\nu}_{12})^2 + \bar{\nu}_{34}^2}\, Z_3\right],\\
P_7 &= Z_1^- Z_2^+ \frac{1}{2|\bar{\nu}_{34}|}\left[|\bar{\nu}_{34}| - (1+\bar{\nu}_{12})X_3 + \sqrt{-(1+\bar{\nu}_{12})^2 + \bar{\nu}_{34}^2}\, Z_3\right],\\
P_8 &= Z_1^+ Z_2^- \frac{1}{2|\bar{\nu}_{34}|}\left[|\bar{\nu}_{34}| - (1+\bar{\nu}_{12})X_3 - \sqrt{-(1+\bar{\nu}_{12})^2 + \bar{\nu}_{34}^2}\, Z_3\right].
\end{aligned}$$

$$P_1 + P_2 = \frac{1}{4}(1+Z_1Z_2)(1-X_3), \qquad P_3 + P_4 = \frac{1}{4}(1+Z_1Z_2)(1+X_3).$$

$$\begin{aligned}
P_5 + P_6 &= \frac{1}{4|\bar{\nu}_{34}|}\left[(1-Z_1Z_2)|\bar{\nu}_{34}| + (1-Z_1Z_2)X_3(1+\bar{\nu}_{12}) + (Z_1-Z_2)Z_3\sqrt{\bar{\nu}_{34}^2 - (1+\bar{\nu}_{12})^2}\right],\\
P_7 + P_8 &= \frac{1}{4|\bar{\nu}_{34}|}\left[(1-Z_1Z_2)|\bar{\nu}_{34}| - (1-Z_1Z_2)X_3(1+\bar{\nu}_{12}) - (Z_1-Z_2)Z_3\sqrt{\bar{\nu}_{34}^2 - (1+\bar{\nu}_{12})^2}\right].
\end{aligned}$$

It is easily verified from the above that

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

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

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

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

Moreover, it is worth noting that Eq. (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 $J_{23}^{zz} = J_{13}^{zz}$ will lead to solutions such as (note the use of (Z_1+Z_2) terms here, making this solution not derivable from Eq. (S13)):

$$\begin{aligned}
\tilde{\mathcal{H}}_{\text{Toff}} = \frac{9\pi}{8} - \frac{7\pi}{8}X_3 + \frac{\sqrt{15}\pi}{8}Z_3 + \frac{\pi}{8}Z_1Z_2\\
\frac{\pi}{8}(Z_1+Z_2)\left(-1 + \frac{5}{2}X_3 + \frac{\sqrt{15}}{2}Z_3\right).
\end{aligned} \tag{S30}$$

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

We will here show a line of thought that could have conceivably led to Eq. (S22) (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 $Z_1 \pm Z_2$:

$$\begin{aligned}
Z_2 + Z_1 &= \text{diag}(2,2,0,0,0,0,-2,-2),\\
Z_2 - Z_1 &= \text{diag}(0,0,-2,-2,2,2,0,0).
\end{aligned} \tag{S31}$$

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

$$\mathcal{H}_1 = -\pi\left(\frac{Z_1+Z_2}{2}\right)\left(\frac{1-X_3}{2}\right), \tag{S32}$$

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 1-qubit interaction term:

$$\begin{aligned}\mathcal{H}_2 &= \frac{1}{2}\left[\mathcal{H}_1 + \frac{\pi}{2}(1 - X_3)\right] \\ &= \pi\,\mathrm{diag}(0, X^-/2, X^-/2, X^-),\end{aligned} \tag{S33}$$

where we remember that $\exp(i\pi X^-) = X$. Equation (S33) 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 normalised 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 Eq. (S33) into something like this we observe that we can rewrite the second term in the above equation as

$$\pi/4(1 - X_3) = \pi/8(2 - 2X_3) = \pi/8(5 - 3 - 2X_3). \tag{S34}$$

Remembering that $Z_1Z_2 = \mathrm{diag}(1, -1, -1, 1)$, we substitute the above with $\pi/8(5 - 3Z_1Z_2 - 2X_3)$. This change affects only the central terms, converting the expression into: $\pi\,\mathrm{diag}(0, 1 - X/4, 1 - X/4, X^-)$. 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

$$\pi\,\mathrm{diag}(0, 1 - X/4 - 2\alpha Z, 1 - X/4 + 2\alpha Z, X^-). \tag{S35}$$

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

$$\begin{aligned}8/\pi\ \mathcal{H}_3 &= -(Z_1 + Z_2)(1 - X_3) \\ &+(5 - 3Z_1Z_2 - 2X_3) \pm \sqrt{15}(Z_2 - Z_1)Z_3.\end{aligned} \tag{S36}$$

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.

II. SUPERVISED LEARNING APPROACH

We here study more in depth the following problem: given a target gate $\mathcal{G}$ and a parametrised 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 $\exp(i\mathcal{H}(\boldsymbol{\lambda}_0)) = \mathcal{G}$? We present a supervised learning approach to numerically solve this problem, considerably extending the ideas presented in Ref. [20]. Thanks to a number of numerical optimisation techniques, and in particular the use of Automatic Differentiation (AD) [37–40], we can explore a variety of different scenarios, optimising over potentially hundreds of Hamiltonian parameters. On top of this, condition (1b) in the main text is used to further speed-up the numerical training, removing many interaction parameters that are known not to lead to the target gate.

A. Supervised learning

Supervised learning is the task of inferring or approximating a function, given a set of pre-labeled data [31, 41]. A supervised learning algorithm starts with some *model* — a functional relation g_λ parametrised by a set of parameters $\boldsymbol{\lambda}$ — and finds a $\boldsymbol{\lambda}_0$ making g_{λ_0} as close as possible to a target function f. To do this, a set of pre-labeled *training data* $\{(x_1, y_1), (x_2, y_2), ...\}$ 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) [42, 43]. These are parametric non-linear models which play a prominent role in many machine learning tasks, such as dimensionality reduction, classification, and feature extraction [42, 43]. NNs have also recently proven useful for several problems in quantum many-body theory [44–51], quantum compilation [52], quantum stabilizer codes [53] and entanglement quantification [54].

A NN is *trained* by optimising its parameters using a dataset of pre-labelled data. A common way to do this is use variations of a gradient-descent-based technique named Stochastic Gradient Descent (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, $\boldsymbol{\nabla} f(\boldsymbol{x})$). The optimal point $\boldsymbol{x}_{\mathrm{opt}}$ 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 $\boldsymbol{x} \to \boldsymbol{x} - \eta\boldsymbol{\nabla} f(\boldsymbol{x})$, 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 parametrised 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

$$\boldsymbol{w} \to \boldsymbol{w} - \eta\boldsymbol{\nabla}_{\boldsymbol{w}} f(\boldsymbol{x}; \boldsymbol{w}). \tag{S37}$$

While standard gradient descent, being a local optimisation 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 $\{\boldsymbol{x}_1, ..., \boldsymbol{x}_M\}$, and updates the parameters using the averaged gradient: $\boldsymbol{w} \to \boldsymbol{w} - \eta\sum_{k=1}^{M}\boldsymbol{\nabla}_{\boldsymbol{w}} f(\boldsymbol{x}_k; \boldsymbol{w})/M$. 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* [55, 56] techniques.

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

$$\mathcal{F}(\boldsymbol{\lambda}, \psi) \equiv \langle\psi|\mathcal{G}^\dagger \exp(i\mathcal{H}(\boldsymbol{\lambda}))\mathcal{G}|\psi\rangle\,, \tag{S38}$$

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 [27–29]. Standard optimisation 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 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.

A crucial step, efficiency-wise, in gradient descent algorithms, is the evaluation of the gradient. Numerically approximating the gradient, as done in previous works [20], is generally inefficient and scales badly with the number of optimised parameters. Here we will instead make use of the powerful technique of Automatic Differentiation (AD) [38, 40], described in Section II B. AD dramatically improves the training efficiency, allowing to explore a richer variety of circumstances.

B. 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 $O(N_{\boldsymbol{w}}^3)$, with $N_{\boldsymbol{w}}$ the number of parameters to differentiate [31]. This inefficiency can however be avoided using *error backpropagation* via AD. With this technique, the complexity of the gradient evaluation phase can be cut down to $O(N_{\boldsymbol{w}}^2)$ [31]. 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 Fig. S1). 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 Fig. S1. 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

$$\boldsymbol{\nabla} g(\boldsymbol{w}) = \sum_k \partial_k f(\boldsymbol{y}^{(2)}) \boldsymbol{\nabla} f_k^{(2)}(\boldsymbol{y}^{(1)}), \tag{S39}$$

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 $\boldsymbol{\nabla} f_k^{(2)}$, which is written as

$$\boldsymbol{\nabla} f_k^{(2)}(\boldsymbol{y}^{(1)}) = \sum_j \partial_j f_k^{(2)}(\boldsymbol{y}^{(1)}) \boldsymbol{\nabla} f_j^{(1)}(\boldsymbol{w}). \tag{S40}$$

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 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.

C. 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 [33], together with the QuTiP library for the simulation of the dynamics of quantum systems [57, 58].

Our implementation allows the training of an arbitrary target gate, parametrised via a time-independent Hamiltonian $\mathcal{H}(\boldsymbol{\lambda})$. The parametrisation 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 parametrisation 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* [55, 56] involves using a modified

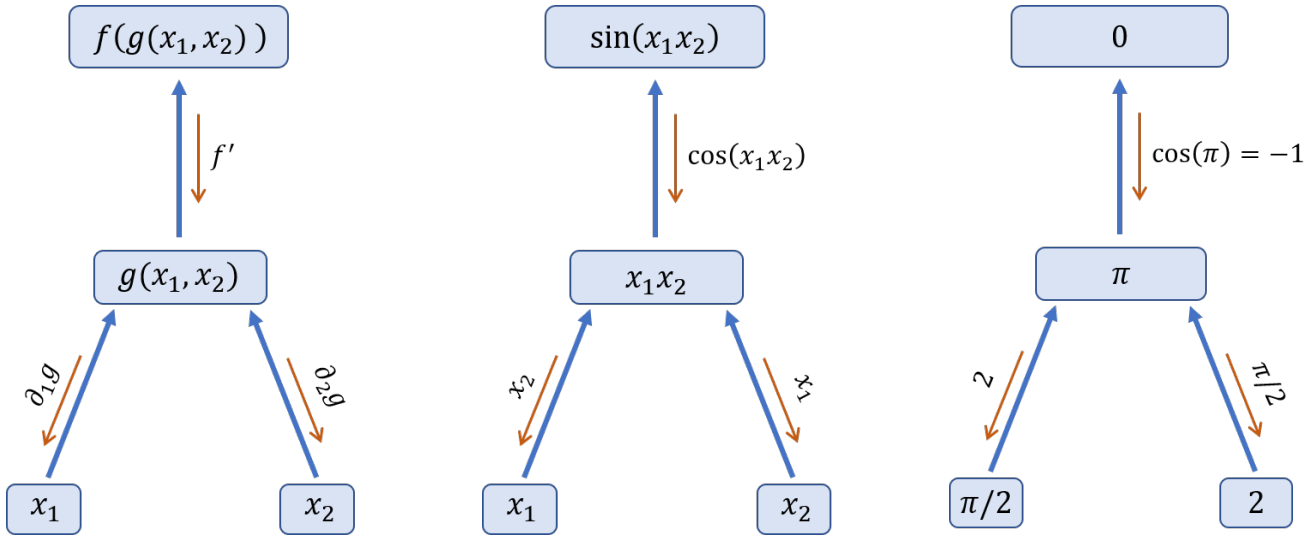

FIG. S1. Examples of AD in backpropagation mode. **(a)** Schematic representation of AD of a function with one output and two inputs. Starting from numerical values for x_1 and x_2, one computes $g(x_1, x_2)$ and then $f(g(x_1, x_2))$. To get $\boldsymbol{\nabla} f(g(x_1, x_2))$, one then computes $f'(g(x_1, x_2))\partial_i g(x_1, x_2)$. 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 $\cos(x_1x_2)$. **(c)** Using the same example function as (b), we give an example of the actual number computed at all stages when the inputs are $(x_1, x_2) = (\pi/2, 2)$.

version of Eq. (S37). The updating rule is instead given by

$$\begin{aligned} \boldsymbol{v} &\to \gamma \boldsymbol{v} + \eta \boldsymbol{\nabla}_{\boldsymbol{\lambda}} \mathcal{F}(\psi, \boldsymbol{\lambda}), \\ \boldsymbol{\lambda} &\to \boldsymbol{\lambda} + \boldsymbol{v}, \end{aligned} \tag{S41}$$

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 [55].

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 [59]. 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 $A \mapsto \Re\mathrm{e}(A) \equiv \mathbb{1} \otimes A_R - i\sigma_y \otimes A_I$, 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 $\Psi \mapsto \Re\mathrm{e}(\Psi) \equiv (\Psi_R, \Psi_I)^T$. It is easy to verify that with this mapping $A\Psi \mapsto \Re\mathrm{e}(A\Psi) = \Re\mathrm{e}(A)\Re\mathrm{e}(\Psi)$, 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:

1. 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.

2. 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.

   (a) Pick N_b of the N_e training states.

   (b) Forward-propagate each state of the sample, and then backpropagate the gradients, thus computing the average gradient over the mini-batch $\boldsymbol{\nabla}_{\boldsymbol{\lambda}} \mathcal{F}(\boldsymbol{\lambda})$.

   (c) Update the coupling strengths λ as per Eq. (S41).

   (d) Return to point (a).

D. Results

A sample of training results for Toffoli, Fredkin, and "double Fredkin" gates are given in Fig. 1 (a), (b), and (c) in the main text. In Figs. S2 to S4 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 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 optimisation hyperparameters are found to give acceptable solutions. For the

trainings shown in this paper we used a dynamically updated learning rate given, for the k^{th} epoch, by $\eta = 1/(1 + k\alpha)$ with the *decay rate* $\alpha = 0.005$. The other hyperparameters were chosen as $\gamma = 0.5$, $N_b = 2$, $N_{tr} = 200$. 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 Figs. S5 to S7 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.

Additional solutions, as well as the exact numbers characterising such solutions, can be found on the GitHub repository lucainnocenti/quantum-gate-learning-1803.07119. This repository contains all the code used to reproduce the solutions presented in this paper, as well as to train arbitrary gates on arbitrary numbers of qubits. Even more generally, arbitrary (linearly) parametrised matrices can be used as training model, allowing a high degree of flexibility.

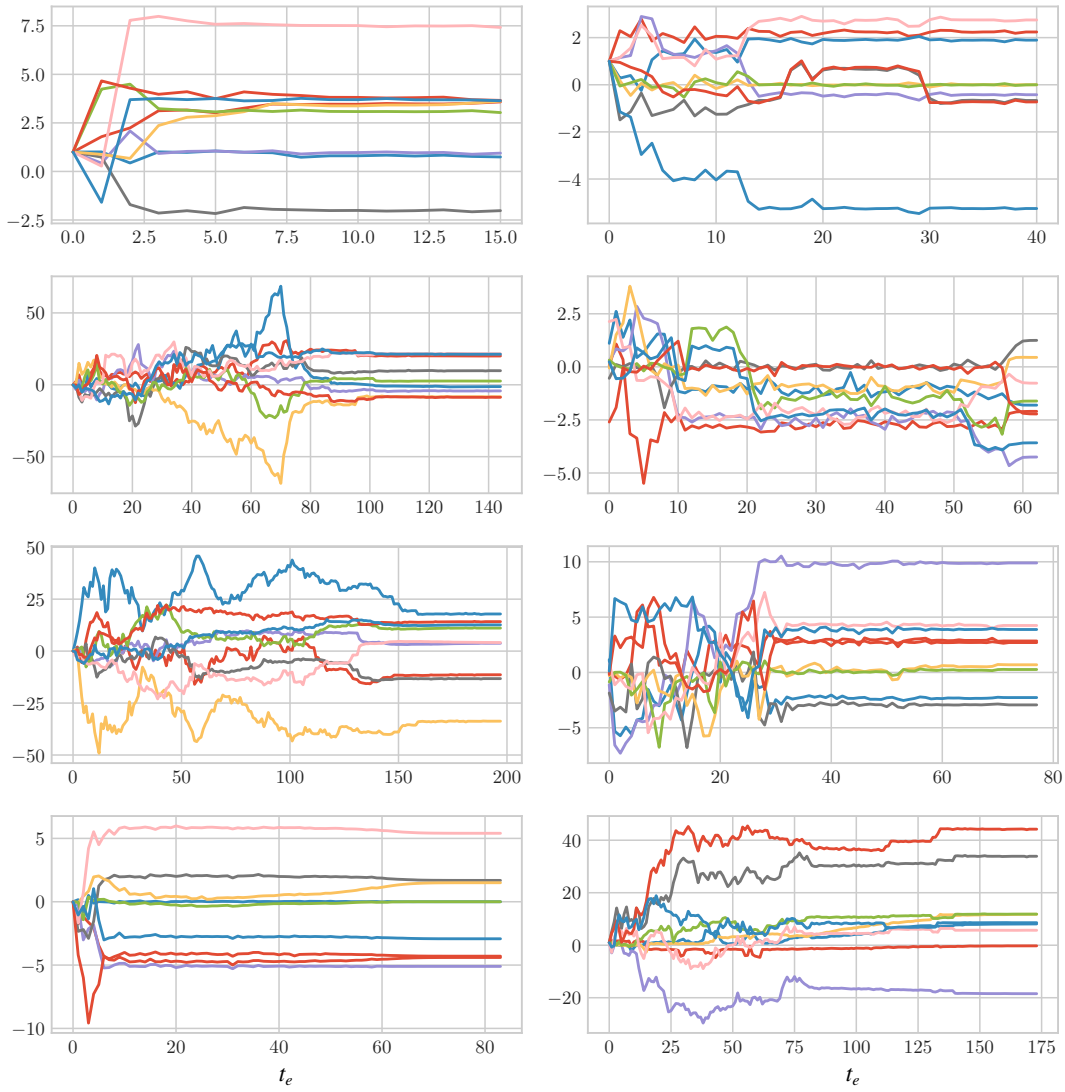

FIG. S2. Training histories for the **Toffoli** gate with only diagonal interactions. In each plot are reported the values of the 9 network parameters during the training process, for each training epoch t_e. Each training process was left running until convergence to unit fidelity, therefore, the number 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.

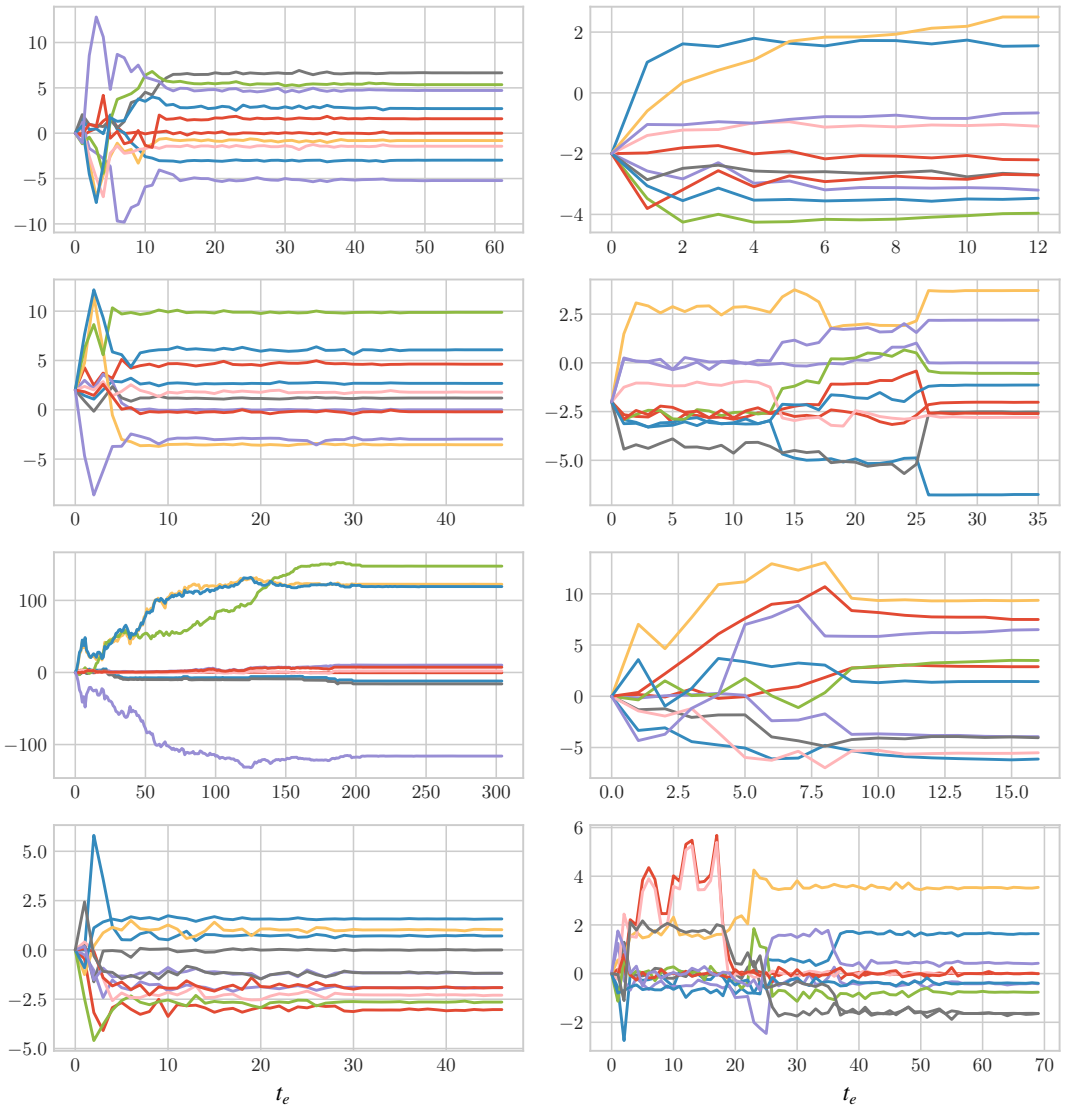

FIG. S3. Training histories for the **Fredkin** gate with only diagonal interactions. In each plot are reported the values of the 9 network parameters during the training process, for each training epoch t_e. Each training process was left running until convergence to unit fidelity, therefore, the number 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.

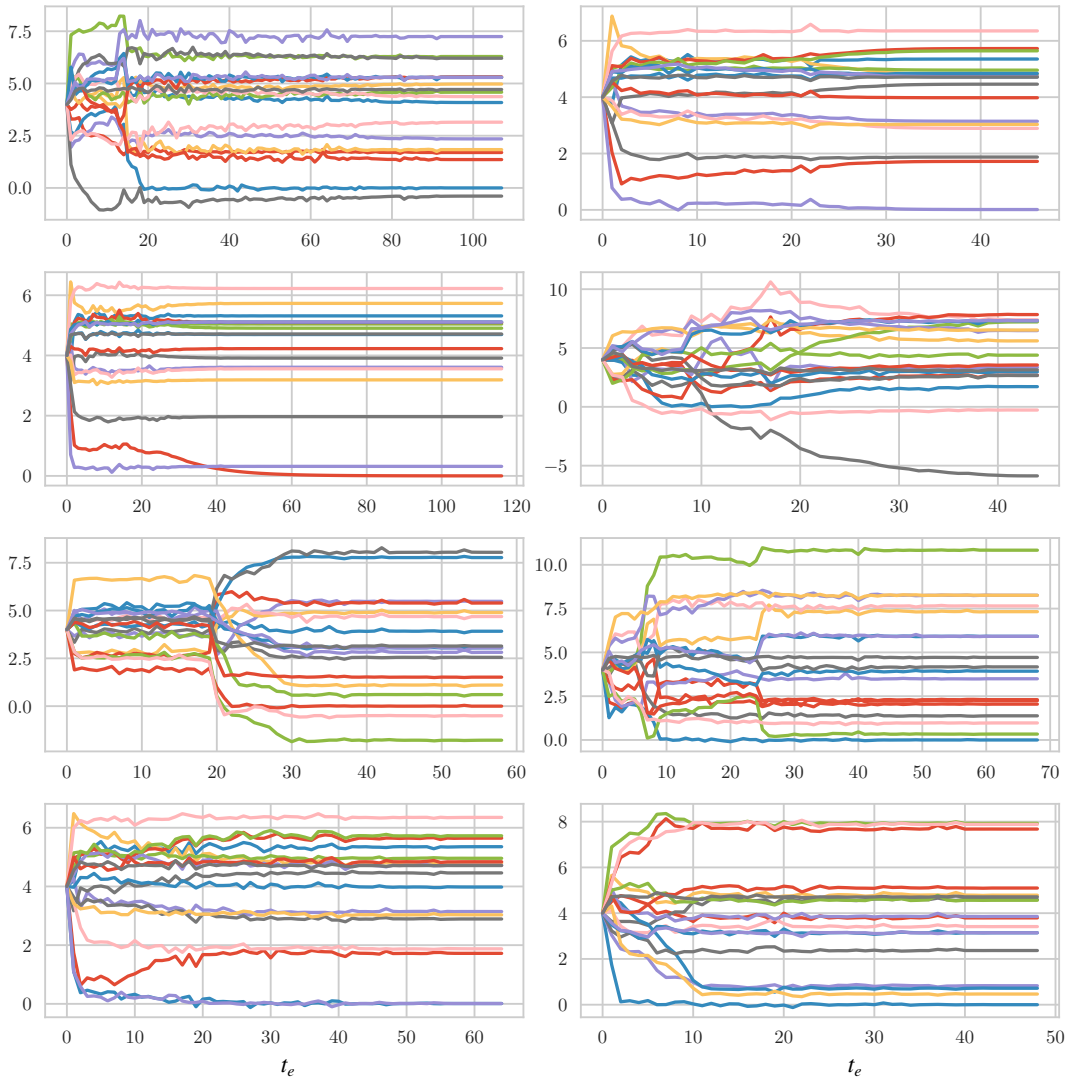

FIG. S4. Training histories for the **double-Fredkin** gate with only diagonal interactions. In each plot are reported the values of the 18 network parameters during the training process, for each training epoch t_e. Each training process was left running until convergence to unit fidelity, therefore, 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 4.

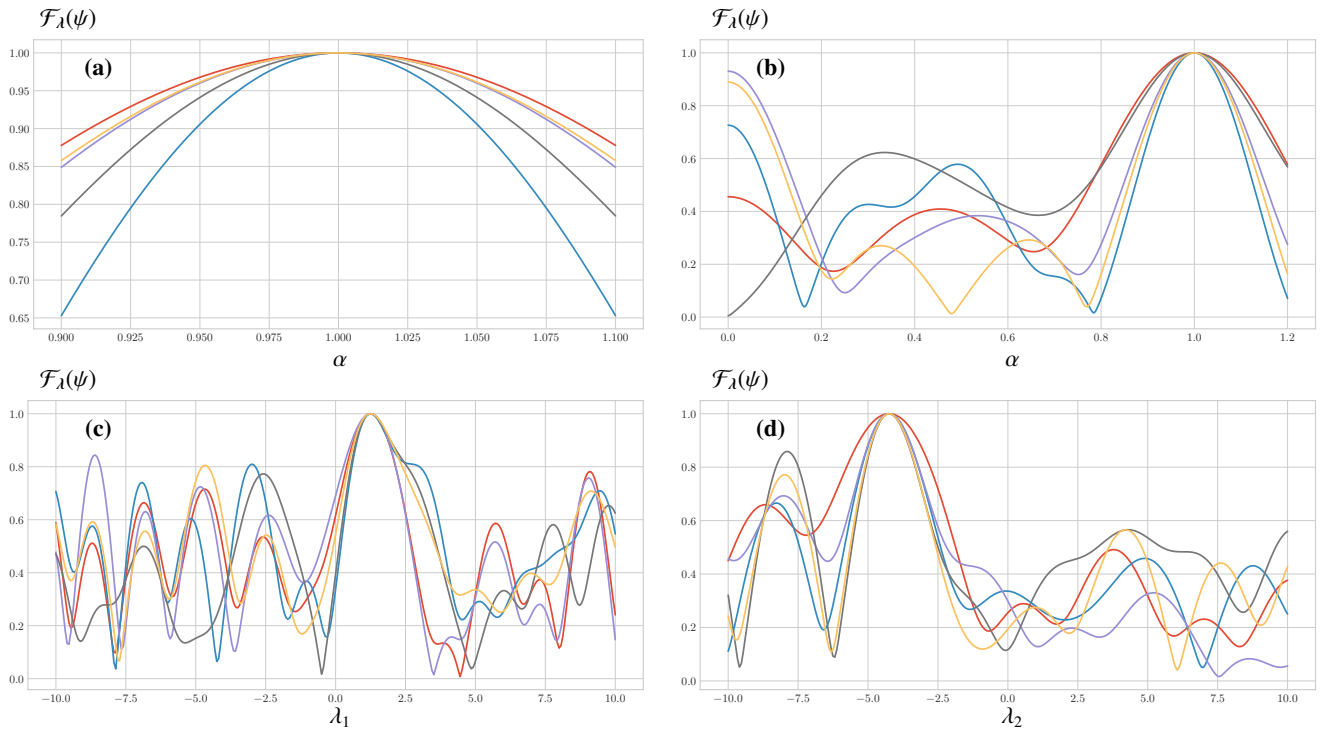

FIG. S5. Fidelity $\mathcal{F}_{\lambda}(\psi)$ vs variations of $\boldsymbol{\lambda}$, for different test states, for the **Toffoli** gate. The five test states ψ are sampled randomly. **(a)** Global relative variations of $\boldsymbol{\lambda}$, that is, plotting the fidelity against $\alpha\boldsymbol{\lambda}$ for $0.9 \leq \alpha \leq 1.1$. 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 $0 \leq \alpha \leq 1.2$. **(c)** Plot of $\mathcal{F}_{\lambda}(\psi)$ 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.

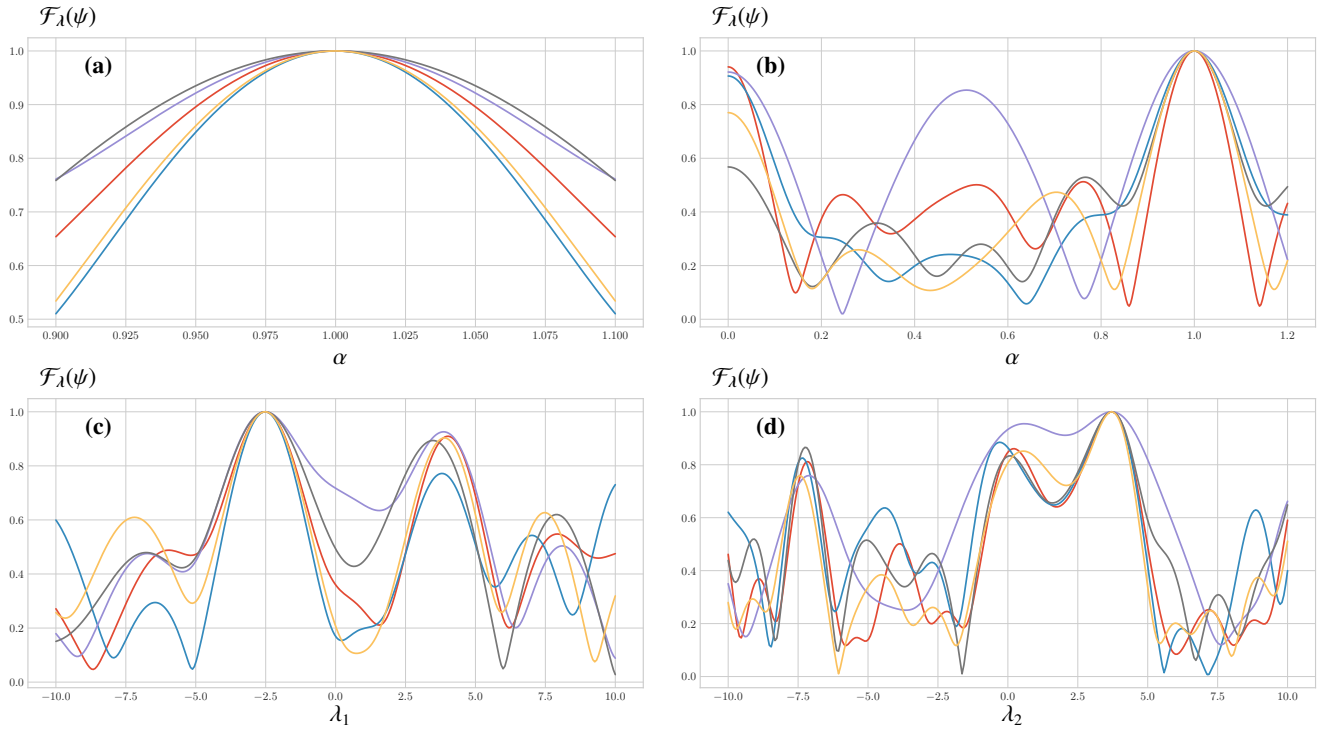

FIG. S6. Fidelity $\mathcal{F}_{\lambda}(\psi)$ vs variations of $\boldsymbol{\lambda}$, for different test states, for the **Fredkin** gate. The five test states ψ are sampled randomly. **(a)** Global relative variations of $\boldsymbol{\lambda}$, that is, plotting the fidelity against $\alpha\boldsymbol{\lambda}$ for $0.9 \leq \alpha \leq 1.1$. 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 $0 \leq \alpha \leq 1.2$. **(c)** Plot of $\mathcal{F}_{\lambda}(\psi)$ 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.

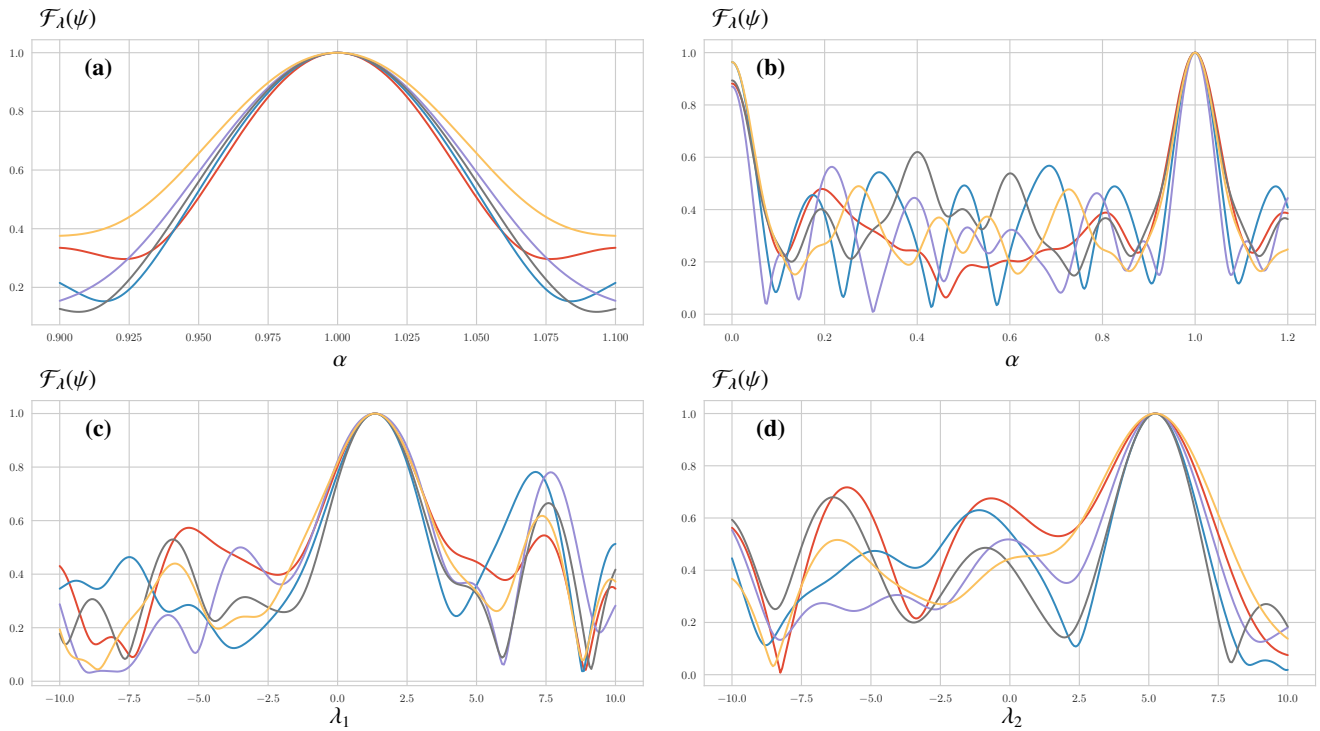

FIG. S7. Fidelity $\mathcal{F}_{\lambda}(\psi)$ vs variations of $\boldsymbol{\lambda}$, for different test states, for the **double Fredkin** gate. The five test states ψ are sampled randomly. **(a)** Global relative variations of $\boldsymbol{\lambda}$, that is, plotting the fidelity against $\alpha\boldsymbol{\lambda}$ for $0.9 \leq \alpha \leq 1.1$. 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 $0 \leq \alpha \leq 1.2$. **(c)** Plot of $\mathcal{F}_{\lambda}(\psi)$ 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.