A differentiable programming method for quantum control

MACHINE
LEARNING
Science and Technology

OPEN ACCESS

RECEIVED
19 February 2020

REVISED
6 May 2020

ACCEPTED FOR PUBLICATION
29 May 2020

PUBLISHED
11 August 2020

PAPER

A differentiable programming method for quantum control

Frank Schäfer, Michal Kloc, Christoph Bruder and Niels Lörch

Department of Physics, University of Basel, Klingelbergstrasse 82, CH-4056 Basel, Switzerland

E-mail: frank.schaefer@unibas.ch **and** michal.kloc@unibas.ch

Keywords: scientific machine learning, differentiable programming, quantum control, quantum parametric oscillator, GHZ states

Abstract

Optimal control is highly desirable in many current quantum systems, especially to realize tasks in quantum information processing. We introduce a method based on differentiable programming to leverage explicit knowledge of the differential equations governing the dynamics of the system. In particular, a control agent is represented as a neural network that maps the state of the system at a given time to a control pulse. The parameters of this agent are optimized via gradient information obtained by direct differentiation through both the neural network *and* the differential equation of the system. This fully differentiable reinforcement learning approach ultimately yields time-dependent control parameters optimizing a desired figure of merit. We demonstrate the method's viability and robustness to noise in eigenstate preparation tasks for three systems: a single qubit, a chain of qubits, and a quantum parametric oscillator.

1. Introduction

In many applications, effective manipulation of physical systems requires an optimization of the available control parameters such as control fields. Since classical intuition often fails for quantum mechanical systems, devising optimal control strategies can be very challenging [1]. A wide range of traditional optimization methods obtain the optimal control pulse sequences by a gradient-based maximization of a certain, task-specific objective [2–7]. Additional constraints beside the main objective can be efficiently implemented using automatic differentiation (AD) [8] and take advantage of GPU acceleration.

In recent years, astonishing advances in reinforcement learning (RL) has led to great opportunities for control optimization. In model-free RL, the training of the agent is based purely on collecting rewards and subsequent modification of network parameters to maximize the expected reward over all possible trajectories without any explicit representation of the system [9]. In contrast to standard optimal control algorithms, perturbations of the initial state are naturally captured within this framework.

Successful demonstrations of model-free RL in physics include, e.g. the manipulation of a quantum-spin chain [10], the inversion of the quantum Kapitza oscillator [11], and quantum gate-control optimization [12]. The authors of reference [10] also show that the performance is comparable to standard optimal control algorithms. By implementation of physical knowledge through a sophisticated reward scheme a student/teacher network could discover quantum error correction strategies from scratch [13].

On the contrary, model-based RL approaches learn an explicit representation of the system, *i.e.* a function which explicitly models state transitions and rewards, simultaneously with a value function or policy. The model of the environment is typically built from the (potentially sparse) reward signals which is, however, computationally expensive. Nevertheless, if feasible, model-based RL bears the potential to learn optimal policies from a smaller number of environment interactions and efficiently handle changing objectives as well [14, 15]. For further reading on different approaches within the RL framework, see, for example, OpenAI's Spinning Up project [16].

Deep-learning methods model the response of a control agent with highly parametrized functions in the form of neural networks (NNs). A useful search direction for the optimization is then indicated by the gradients of these parameters with respect to a given figure of merit. If the environment is implemented in a computer simulation where primitives for all derivative operations are defined [17], AD allows us to backpropagate directly through the time trajectory and we thus effectively switch from model-free to model-based RL [18, 19]. Given such a differentiable simulation, a great speed-up in the optimization process was demonstrated [20, 21]. This approach is part of a paradigm named differentiable programming (DP) where programs are designed in a fully differentiable way [22, 23]. The learnable structures, such as NNs, are then embedded in a standard procedural code as a way to obtain the resulting program. Among the first examples, it was demonstrated that DP allows one to find optimal solutions much faster than model-free RL for standard classical problems [18, 22]. In physics, tensor network algorithms were programmed in a fully differentiable way [24].

In this paper, we follow this pivotal shift in the usage of RL and show that using the DP method, a control agent can be trained to provide successful control strategies for noisy inputs within different quantum mechanical systems. We start by introducing and illustrating the workflow of our method using the most basic quantum system of a single qubit in section 6.1. We then evaluate the method on two further examples: In section 6.2, we investigate the preparation of the GHZ state in a chain of qubits. In section 6.3, the eigenstate preparation in the case of a quantum parametric oscillator is considered.

2. Quantum control problem

Suppose a quantum system is described by a Hamiltonian

$$H(t) = H_0 + \sum_{k=1}^{K} u_k(t) H_k\,, \tag{1}$$

where H_0 is a time-independent part (which we will refer to as the drift term) and H_k represents control fields with respective scalar amplitudes $u_k(t)$. The total number of independent control fields is K. The dynamics of a state $|\psi(t)\rangle$, represented as a vector in a Hilbert space, is governed by the Schrödinger equation $\mathrm{i}\partial_t|\psi(t)\rangle = H(t)|\psi(t)\rangle$ given some initial state $|\psi(t_0)\rangle \equiv |\psi(t=0)\rangle$ (we use $\hbar = 1$ throughout the paper). The goal of quantum control is to optimize $u_k(t)$ over a given time interval according to a certain figure of merit, e.g. to maximize an overlap with some target state. The overlap $F(t) = |\langle\psi(t)|\psi_{\mathrm{tar}}\rangle|^2$ is generally called fidelity[1].

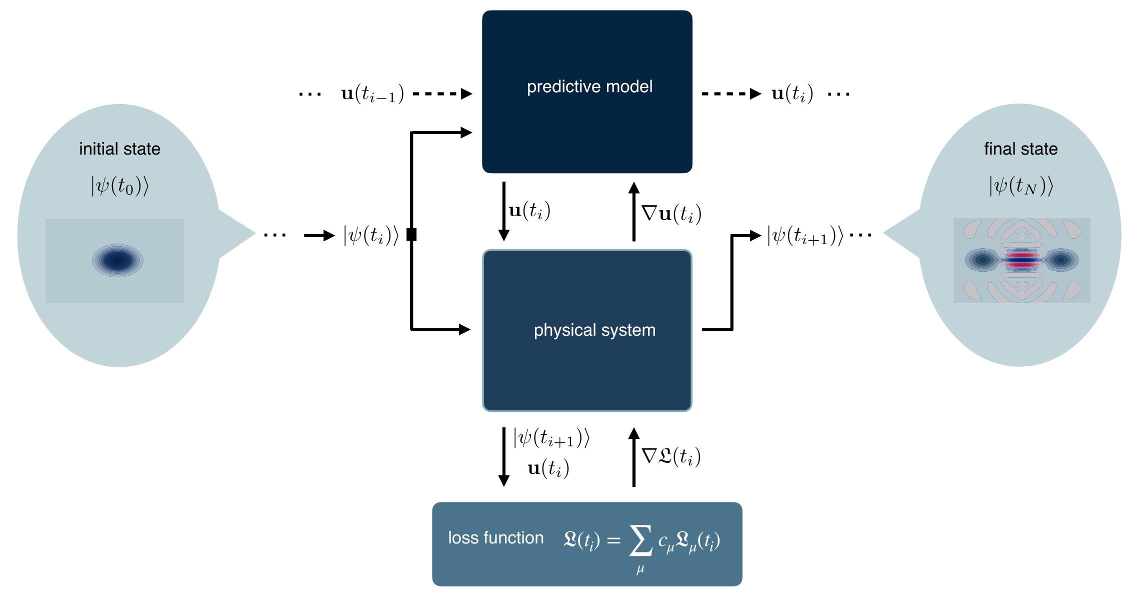
Many numerical approaches consist of discretizing the time interval into N steps in which the control amplitudes are considered constant. The pulse sequences $u_k(t_i)$, $i = 1\ldots N$ are then optimized by gradient methods. Popular examples are the GRAPE algorithm [4] and the class of algorithms generally attributed to Krotov [3]. Alternative approaches like the CRAB algorithm [25, 26] do not rely on evaluating gradients but rather map the problem to finding the extrema of a multivariate function using direct search methods such as Simplex methods [25].

A common feature of the state preparation methods listed above is their sensitivity to the input state. If the input state is altered, the control fields will in general no longer be optimal. It should be noted, though, that proposals for generalization of the approach have been already put forward, see references [5, 27, 28]. RL formulates control tasks in terms of Markov Decision Processes which map an individual state $|\psi(t_i)\rangle$ to the associated actions. Therefore, the noisy inputs can be treated very naturally, provided that the agent was exposed to sufficiently many training examples. Our differentiable programming method, as described in the next section, uses a similar strategy.

3. Quantum optimal control with differentiable programming

Let us now describe how a differentiable programming approach allows us to solve the quantum optimal control problem as defined in section 2. The method consists of three main building blocks: the predictive model, the physical system, and the loss function $\mathfrak{L}$. The connection between these building blocks is depicted in figure 1. Starting from some initial state, $|\psi(t_0)\rangle$, and vanishing control fields, in each time step i the predictive model receives the current quantum state, $|\psi(t_i)\rangle$, as well as the last control field strengths, $u_k(t_{i-1})$, as an input. The predictive model maps this input to the next control field strengths, $u_k(t_i)$. Adopting the terminology of RL, we now also refer to the predictive model as *agent* and to the control fields as *actions*. According to our experience, if the agent is aware of its last actions, the optimal strategy can be found more easily and often provides smoother control signals. Moreover, the output of the predictive model can be easily reformulated to return the gradients rather than the action themselves. In that case, the maximal fluctuations of the control fields between the time steps can be naturally controlled which may be required in experimental applications. Solving the ordinary differential equation (ODE) of the physical

[1]Let us note that a control problem can be formulated more generally, e.g. also as a unitary operator synthesis or within the context of dissipative dynamics. In the present work, we restrict our investigations to the state preparation and unitary time evolution.



predictive model
$\cdots$ $\mathbf{u}(t_{i-1})$
$\mathbf{u}(t_i)$ $\cdots$
initial state
$|\psi(t_0)\rangle$
$\cdots$ $|\psi(t_i)\rangle$
$\mathbf{u}(t_i)$
$\nabla\mathbf{u}(t_i)$
$|\psi(t_{i+1})\rangle \cdots$
final state
$|\psi(t_N)\rangle$
physical system
$|\psi(t_{i+1})\rangle$
$\mathbf{u}(t_i)$
$\nabla\mathfrak{L}(t_i)$
loss function $\mathfrak{L}(t_i) = \sum_\mu c_\mu \mathfrak{L}_\mu(t_i)$

Figure 1. Sketch of our differential programming approach for quantum optimal control. In the forward pass, starting from an initial state, $|\psi(t_0)\rangle$ (and vanishing control fields), in each time step i, the predictive model maps the quantum state $|\psi(t_i)\rangle$ to the associated optimal control amplitudes $\mathbf{u}(t_i)$. These amplitudes determine the time evolution of the physical system from $|\psi(t_i)\rangle$ to $|\psi(t_{i+1})\rangle$, as computed by an ODE solver. Both the state $|\psi(t_{i+1})\rangle$ and the amplitudes $\mathbf{u}(t_i)$ contribute to the loss function $\mathfrak{L}(t_i)$. This procedure is iterated for all N steps. In the backward pass, automatic differentiation allows us to compute the gradients of $\mathfrak{L}(t_i)$ with respect to the parameters of the predictive model by direct differentiation through the ODE solver. These gradients are used to optimize the model's parameters in a series of epochs. To ensure sufficiently smooth control fields and to achieve a faster learning progress, we also give the last amplitudes $\mathbf{u}(t_{i-1})$ to the predictive model as an input (dashed arrows).

system, as determined by the Hamiltonian and the Schrödinger equation, leads to the next state $|\psi(t_{i+1})\rangle$. This state, as well as the actions taken, $u_k(t_i)$, enter the computation of the loss function which maps the inputs to a scalar value. This procedure is iterated for all time steps.

Figure 2 shows the architecture of the predictive model. We use artificial NNs with linear (fully-connected) layers and rectified linear units (ReLUs) as the activation functions. We divide the full NN into three separate networks:

(i) A *state-aware* network f_s that takes the current quantum state, $|\psi(t_i)\rangle \in \mathbb{C}^D$, and computes a map

$$f_s(|\psi(t_i)\rangle) : \mathbb{C}^D \to \mathbb{R}^B, \tag{2}$$

where D is the Hilbert space dimension and B is the number of output features in the final layer of the state-aware network.

(ii) An *action-aware* network f_a that takes the last actions, $u_k(t_{i-1}) \in \mathbb{R}^K$, and computes a map

$$f_a(u_k(t_{i-1})) : \mathbb{R}^K \to \mathbb{R}^B, \tag{3}$$

where K is the total number of control fields and B is identical as above, and

(iii) a *combination-aware* network f_c that post-processes the sum of the two maps $f_s(|\psi(t_i)\rangle) + f_a(u_k(t_{i-1}))$, and predicts the next control fields

$$f_c(f_s(|\psi(t_i)\rangle) + f_a(u_k(t_{i-1}))) : \mathbb{R}^B \to \mathbb{R}^K, \tag{4}$$

with B and K as above.

Here, we combined f_s and f_a by a simple addition, but other combinations such as concatenation are also possible. Which combination is most efficient in a given situation will depend on the particular physical system and number of parameters of the model.

All our code is implemented in PyTorch [29] which has no complex tensor support, yet. Therefore, we map the Schrödinger equation via the isomorphism

$$\partial_t|\psi(t)\rangle = -\mathrm{i}H(t)|\psi(t)\rangle \mapsto \partial_t \begin{bmatrix} |\psi_{\mathrm{Re}}(t)\rangle \\ |\psi_{\mathrm{Im}}(t)\rangle \end{bmatrix} = \begin{bmatrix} H_{\mathrm{Im}}(t) & H_{\mathrm{Re}}(t) \\ -H_{\mathrm{Re}}(t) & H_{\mathrm{Im}}(t) \end{bmatrix} \begin{bmatrix} |\psi_{\mathrm{Re}}(t)\rangle \\ |\psi_{\mathrm{Im}}(t)\rangle \end{bmatrix} \tag{5}$$

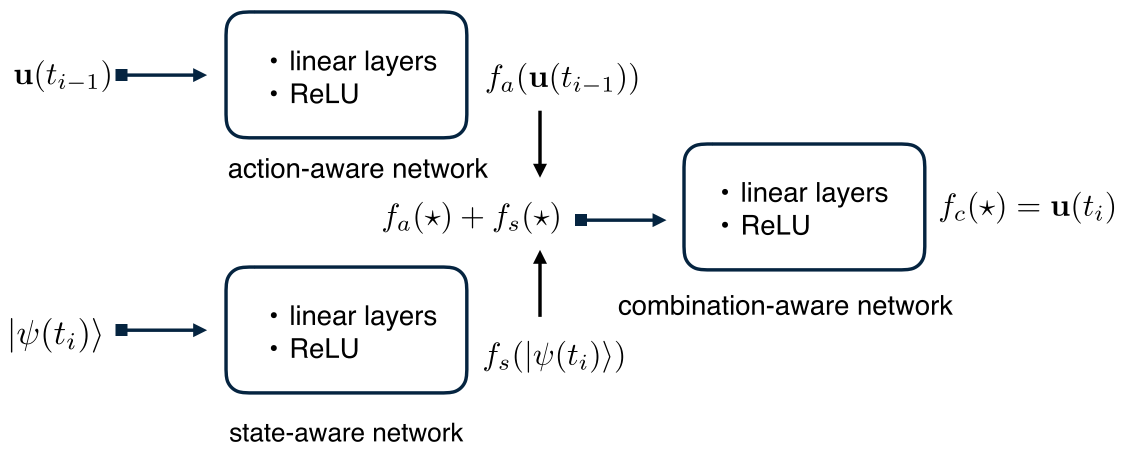


Figure 2. The general neural network architecture for the predictive model used throughout all our numerical experiments. The number of weights and layers used for each network is given in table A1 in appendix A.

to real-valued vectors and matrices. The subscripts Re/Im denote the real and imaginary part, respectively. Similarly, $|\psi(t_i)\rangle$ is mapped onto $[|\psi_{\mathrm{Re}}(t_i)\rangle, |\psi_{\mathrm{Im}}(t_i)\rangle]^T$ as the input for the state-aware network. For the time evolution of the physical system, as described by the ODE problem (5), we use Heun's method [30] to get the next time point $i+1$.

The subsequent computation of the loss function consists of a weighted sum of individual contributions $\mathfrak{L}_\mu$ that have a well-defined physical meaning. These terms are explained in detail in section 5. Importantly, since the physical system is implemented entirely in PyTorch and is differentiable, the derivative of the loss function with respect to the network's weights can be computed at any point in time.

We use the Adam optimizer [31] to train the model in a series of epochs, where b simulations are carried out in parallel per epoch, see figure 2. Hyperparameters, such as the coefficients of the individual loss terms as well as the number of output features of the linear layers, are obtained either empirically or using the Bayesian Optimization and Hyperband method (BOHB) [32] which is a combination of random search (bandit-based method) with Bayesian optimization.

The embedding of the physical model into the backpropagation pass (which is not the case in model-free RL), has led to far more effective control strategies and faster training for classical environments [18]. The potential robustness of the method with respect to noise in the input is desired for practical purposes where inputs can not be prepared to arbitrary precision. To test the performance of our scheme, we apply it to different physical systems, which are introduced in the next section.

4. Physical systems

4.1. Qubit chain with nearest-neighbor interaction

We consider a chain of M identical qubits (or spins $\frac{1}{2}$) with the control Hamiltonian

$$H(t) = \sum_{i=1}^{M} J\sigma_z^{(i)}\sigma_z^{(i+1)} + u_x^{(i)}(t)\sigma_x^{(i)} + u_y^{(i)}(t)\sigma_y^{(i)}\,, \tag{6}$$

where $\sigma_x^{(i)}, \sigma_y^{(i)}, \sigma_z^{(i)}$ are the Pauli matrices acting on the ith site of the chain. For any single qubit we have two independent control fields with amplitudes $u_x^{(i)}(t)$ and $u_y^{(i)}(t)$. They allow us to rotate each spin by an arbitrary angle as illustrated in figure 3(a).

The dimension of the Hilbert space scales as $D = 2^M$. The set of vectors representing all possible configurations of the z projection at individual sites forms a basis. Therefore a general state vector $|\psi\rangle$ can be expanded as

$$|\psi\rangle = C_0|\uparrow\uparrow\uparrow..\uparrow\uparrow\rangle_z + C_1|\uparrow\uparrow\uparrow..\uparrow\downarrow\rangle_z + \ldots + C_{D-1}|\downarrow\downarrow\downarrow..\downarrow\downarrow\rangle_z\,, \tag{7}$$

with unique coefficients $C_0, \ldots, C_{D-1}$. The drift term has the form of a nearest-neighbor interaction. As we set $J > 0$, the ground state is given by the doubly degenerate manifold spanned by the Néel states $|\psi^1_{\mathrm{gs}}\rangle = |\uparrow\downarrow\uparrow\ldots\rangle_z$ and $|\psi^2_{\mathrm{gs}}\rangle = |\downarrow\uparrow\downarrow\ldots\rangle_z$.

In section 6.1 we consider a simplified model case with only one qubit and the Hamiltonian

$$H = \frac{\omega}{2}\sigma_z + u_x(t)\sigma_x\,, \tag{8}$$

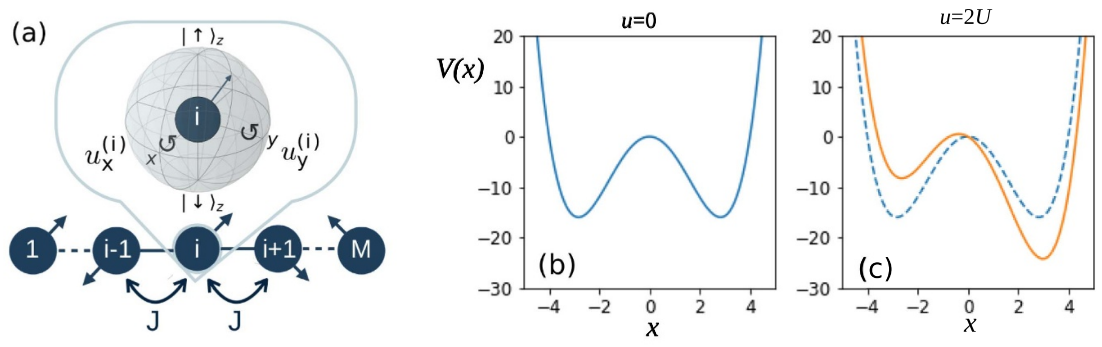


Figure 3. Panel (a): A chain of M qubits and the control fields. According to the Hamiltonian (6), each qubit interacts with its nearest neighbors with strength J. The control fields $u_x^{(i)}$, $u_y^{(i)}$ are applied at each site. Therefore, any qubit can be rotated by an angle prescribed by the respective field strengths. Panels (b) and (c): Shape of the classical potential of the Hamiltonian (9) describing a quantum parametric oscillator. Both cases with the control field u being switched off (panel b) and switched on (panel c) are depicted.

where ω is the qubit frequency. Note that the single control field $u_x(t)$ is sufficient to realize any rotation as σ_y can be obtained as a commutator of σ_x and σ_z.

4.2. Quantum parametric oscillator

The other system considered is a driven non-linear optical resonator whose dynamics can be described by the following Hamiltonian in the rotating frame (see, for example, reference [33])

$$H(t) = Ua^\dagger a^\dagger aa + G\left(a^\dagger a^\dagger + aa\right) + u(t)\left(a + a^\dagger\right). \tag{9}$$

The first term is a Kerr (non-linear) interaction between the photons. They are annihilated and created using the respective operators $a,\ a^\dagger$. We can represent a general state $|\psi\rangle$ in the Fock basis, *i.e.* the basis of photon occupation number states $|n\rangle$, as

$$|\psi\rangle = \sum_{n=0}^{\infty} C_n|n\rangle, \qquad C_n \equiv \langle n|\psi\rangle. \tag{10}$$

The mean photon number in the state $|\psi\rangle$ is given as $\langle n\rangle_\psi = \langle\psi|a^\dagger a|\psi\rangle$.

In the following, we fix the amplitude of the coherent two-photon drive relative to the strength of the Kerr non-linearity $G = -4U$. Our control scheme consists of one tunable parameter u(t) which controls the amplitude of the single-photon drive. Its action can be intuitively understood from the classical counterpart of a similar system. The classical potential $V(x)$ is obtained from equation (9) by writing the operators as c-numbers $\sqrt{2}a = x + ip$ and setting $p = 0$. Figure 3(b) reveals a symmetric double-well potential when the control field is switched off. If the control field is switched on, the potential becomes tilted as shown in figure 3(c).

5. Loss functions

Multiple objectives can be passed on to the agent via a case-specific loss function $\mathfrak{L}$. We assume the generic form [8, 26, 34]

$$\mathfrak{L} = \sum_\mu c_\mu \mathfrak{L}_\mu, \tag{11}$$

which is a linear combination of elementary loss functions $\mathfrak{L}_\mu$ encoding specific details of the control task. The agent will reflect these details according to their relative importance given by the coefficients c_μ. They have to be optimized either empirically (as in reference [8]) or by means of some hyperparameter optimization algorithms.

The elementary loss functions used in this paper are summarized in table 1. The function $\mathfrak{L}_F$ is a discounted sum over the infidelity for every time step i with the real positive discount factor $\gamma \leq 1$. This loss function serves as the main objective for our control problems as its minimization leads to a maximal overlap

Table 1. Elementary loss functions employed for quantum optimal control in our numerical experiments.

primary goal	
target-state infidelity	$\mathfrak{L}_F = \sum_i^N \gamma^i \left(1 - \lvert\langle\psi(t_i)\vert\psi_{\text{tar}}\rangle\rvert^2\right)$
	$\mathfrak{L}_{FN} = \lvert\langle\psi(t_N)\vert\psi_{\text{tar}}\rangle\rvert^2$
constraints	
control amplitudes	$\mathfrak{L}_{\text{amp}} = \sum_i^N \lvert\mathbf{u}(t_i)\rvert$
	$\mathfrak{L}'_{\text{amp}} = \sum_i^N \lvert\mathbf{u}(t_i)\rvert^2$

with the target state on the entire control interval. Therefore, minimization of $\mathfrak{L}_F$ minimizes the time to reach the target state.

In the case of eigenstate preparation, once the target state has been reached, the control fields can be switched off. Then, the state remains unchanged under the time evolution according to the drift Hamiltonian. We add the loss function $\mathfrak{L}_{FN}$ to ensure that the target state is prepared at the final time step. To focus on another particular point in time, $\mathfrak{L}_{FN}$ can be adjusted accordingly.

The remaining loss functions in table 1 represent additional constraints on the final control pulse sequence. Namely, by employing $\mathfrak{L}_{\text{amp}}$ or $\mathfrak{L}'_{\text{amp}}$ the agent is forced to prefer smaller amplitudes[2]. Of course, other elementary loss functions can be added to equation (11) according to the specific requirements of the control task.

6. Numerical experiments

6.1. Optimal control of a single qubit

We start with a minimal quantum model to illustrate the workflow of our method. We consider a qubit in a magnetic field that can be manipulated by a control field in x direction described by the Hamiltonian (8). The general task is to move an arbitrary initial state $\vert\psi(t_0)\rangle = \cos(\frac{\theta}{2})\vert\uparrow\rangle_z + \sin(\frac{\theta}{2})e^{i\phi}\vert\downarrow\rangle_z$, with $\theta \in [0, \pi]$ and $\phi \in [0, 2\pi)$ uniformly sampled on the Bloch sphere, into the state $\vert\psi_{\text{tar}}\rangle = \vert\uparrow\rangle_z$ with high fidelity in the shortest possible time. Note that this task is significantly different from the typical quantum optimal control setup where the initial state has fixed values for θ and ϕ.

The predictive model is constructed as in figure 2 and the number of weights and biases are constant for all qubit experiments, see table A1 in appendix A. We use the Bayesian Optimization and Hyperband (BOHB) method [32] to tune the number of parallel simulations b, the learning rate of the Adam optimizer, and the coefficients c_F, c_{FN} and c'_{amp} of the loss function $\mathfrak{L}$ (11). The objective for BOHB must not depend explicitly on the c_μ, as they are also optimized. A natural choice for the BOHB objective is $\mathfrak{L}_F$ from table 1. More details of the hyperparameter optimization can be found in appendix B.

The results of the control task with the optimized hyperparameters are shown in figure 4. The two rows compare different choices of the number of hyperparameters in the loss function (11). The first two columns visualize the loss and the mean of the final state infidelity as a function of the epochs. Smaller final fidelities are reached when several loss functions are combined. We applied the trained model to a test set of 512 random configurations. The results for the mean and the standard deviation of the fidelity as a function of time and of the applied control fields clearly show the different behaviors for different choices of hyperparameters in the loss function. Particularly, the term $\mathfrak{L}'_{\text{amp}}$ pushes the actions to zero and hence helps to achieve a well-defined shorter control sequence. Comparison with previous results [8] shows that we get a very similar control pulse when the initial state is chosen as the ground state of the drift term of the Hamiltonian (8). A comparison with a RL approach can be found in appendix C.

6.2. GHZ state preparation in a chain of qubits

In this section, we aim at the preparation of a GHZ state [35] in a chain of M qubits

$$\vert\text{GHZ}\rangle = \frac{1}{\sqrt{2}}\left(\vert\uparrow\rangle_z^{\otimes M} + \vert\downarrow\rangle_z^{\otimes M}\right) = \frac{1}{\sqrt{2}}\left(\vert\uparrow\uparrow\ldots\rangle_z + \vert\downarrow\downarrow\ldots\rangle_z\right). \tag{12}$$

These states play an important role, e.g. for multi-particle generalizations of superdense coding [36] or in quantum secret sharing [37].

The initial state $\vert\psi(t_0)\rangle$ is chosen as one of the ground states of the drift Hamiltonian from equation (6) (states $\vert\psi^1_{\text{gs}}\rangle$ and $\vert\psi^2_{\text{gs}}\rangle$, see section 4.1). We assume the input contains noise which is implemented as flipping any single qubit in the initial state with 10% probability.

[2]Here, the difference given by L1 or L2 regularization is purely technical. However, the squared amplitude can be linked with the intensity of the control field which in many cases represents an experimentally more relevant quantity than the amplitude itself.

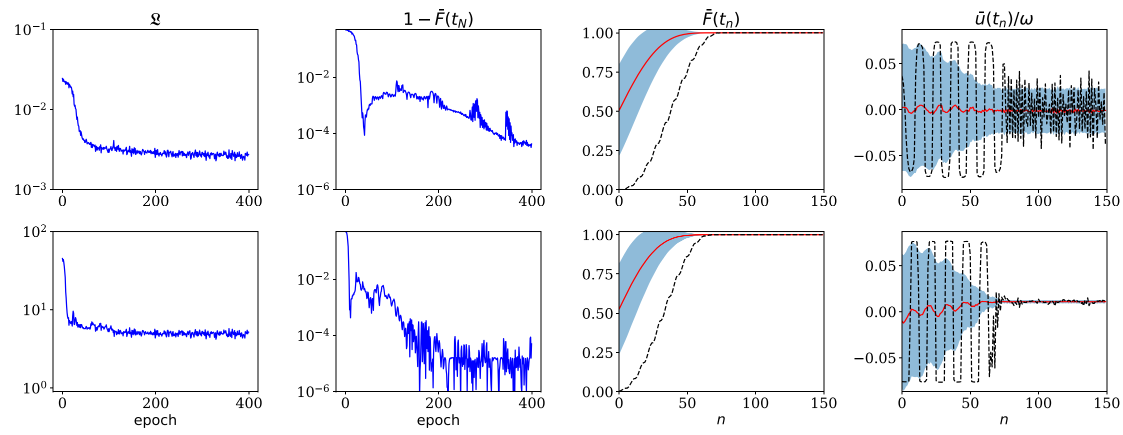


Figure 4. Preparation of the eigenstate $|\uparrow\rangle$ of the drift term of the Hamiltonian (8). The first row shows the results when the loss function $\mathfrak{L}$ (11) contains only one non-zero coefficient c_F. The second row shows the optimized results for three contributing coefficients c_F, c_{FN}, c_{amp}. In the first column, the evolution of $\mathfrak{L}$ during the training phase is shown. The second column shows the mean value of the final state infidelity as a function of the epochs. The third and fourth columns show the performance of the model when it is applied to a test set of size 512. The red curve/ blue shaded region indicate the mean/standard deviation of the fidelity and the actions, respectively. The black dashed line highlights the special case with initial state given as $|\psi(t_0)\rangle = |\downarrow\rangle$. All hyperparameters can be found in table A1 in appendix A.

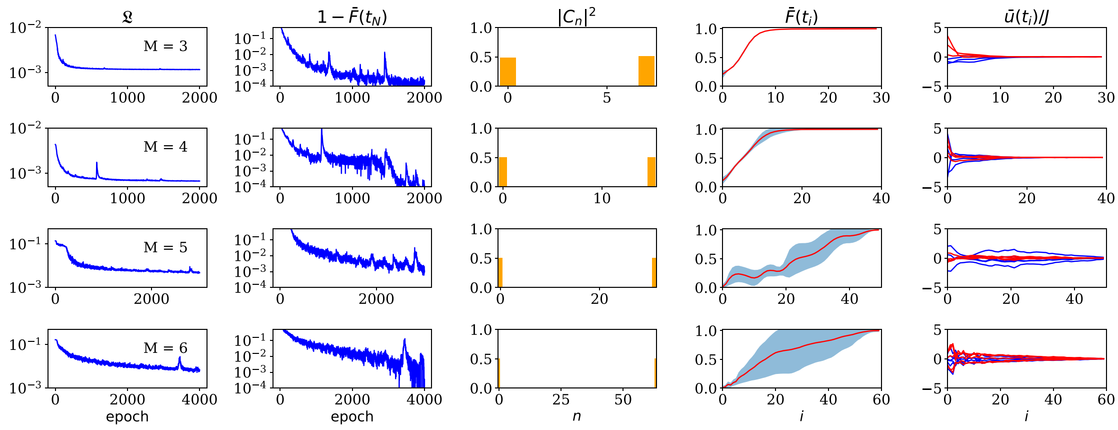


Figure 5. GHZ state preparation for various lengths M of the spin chain. From top to bottom, the number of sites M increases. The first two columns show the progress of the loss function $\mathfrak{L}$ and the final state infidelity averaged over b parallel trajectories $1-\bar{F}(t_N)$ as a function of the training epochs. Columns three to five show results of the trained NN applied to a test set of 256 spin configurations. The coefficients C_n plotted in the third column refer to equation (7). The GHZ state corresponds to equal occupations of the leftmost and the rightmost columns in the histogram, *i.e.* $|C_0|^2 = |C_{D-1}|^2 = 0.5$. The evolution of the mean fidelity $\bar{F}$ of the test set during the preparation process for each step $i = 1 \ldots N$ is plotted in the fourth column. The blue bands highlight one standard deviation. The last column depicts the applied 2 M independent actions averaged over the test set for each step i. All hyperparameters can be found in table A1 in appendix A.

According to our goal, we set $|\psi_{\text{tar}}\rangle = |\text{GHZ}\rangle$. Note that the target is composed of two eigenstates of the drift Hamiltonian with identical eigenvalues. Therefore, once the GHZ state has been prepared, the control fields can be switched off, and the state evolves trivially with the drift Hamiltonian. Hence, the control problem fits well into the category of eigenstate preparation.

The coefficients c_μ from equation (11) were obtained using the BOHB algorithm. The same algorithm was also used to optimize other hyperparameters, *i.e.* the learning rate, the number of parallel trajectories b, and the number of weights in each layer of the NN for any size of the system M. Their values are summarized in table A1 in appendix A.

The training phase of the NN as well as examples of its performance on test data sets are shown in figure 5. As compared to a typical RL method, the loss functions $\mathfrak{L}$ show rapid initial progress and do not display any strong oscillations during the learning process. The performance we require from our control scheme is to reach fidelities above 99.9% at latest at the end of the control interval. Obviously, the number of the training epochs required to accomplish this grows with M (see the second column of figure 5). However, for all system sizes considered we manage to train the NN to reach the desired performance as demonstrated in the third and the fourth columns of figure 5. Indeed, even for the case with $M = 6$ qubits, where the initial noise leads to higher fluctuations in the test data, the final average fidelity $\bar{F}(t_N)$ meets our precision

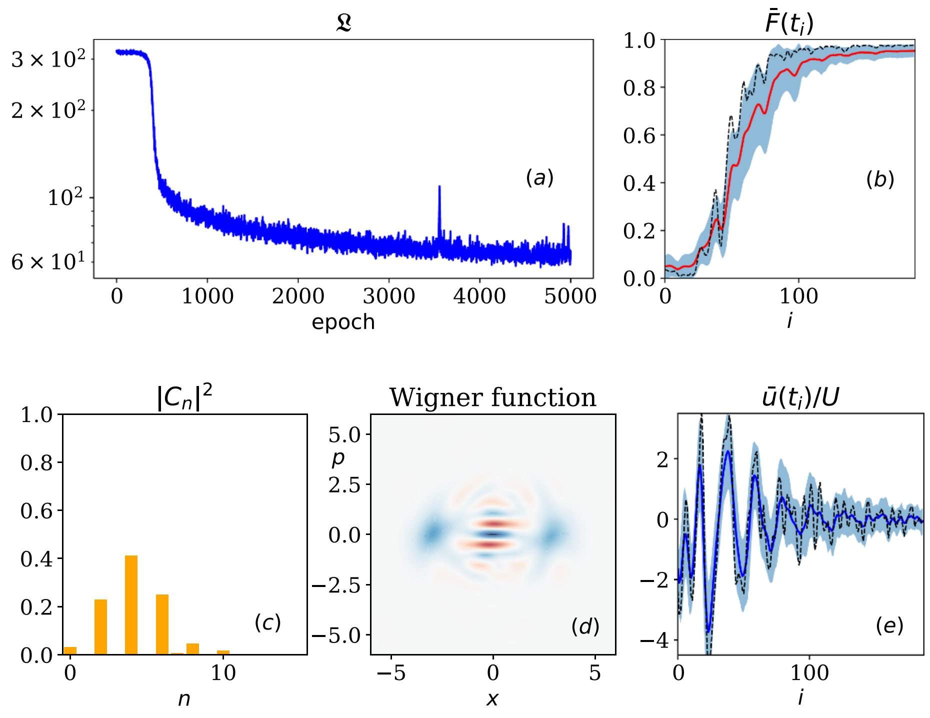


Figure 6. Eigen-cat state preparation in the quantum parametric oscillator. The training of the NN and the performance on a test set with 64 initial states are depicted. The noise bandwidth is $2\xi = 0.8$. Panel (a): Progress of the loss function $\mathfrak{L}$ during the training averaged over b parallel trajectories. Panel (b): Evolution of the mean fidelity $\bar{F}$ of the test set, the blue region shows one standard error. Panels (c) and (d): Fock distribution and the Wigner function of a randomly chosen state from the test set at the final time step t_N. The coefficients C_n refer to equation (10). Panel (e): Mean actions performed by the agent during the test run, the blue region shows one standard error. The black dashed lines in panels (b) and (e) show a the special case with an unperturbed initial vacuum state $|\psi(t_0)\rangle = |0\rangle$. All hyperparameters can be found in table A1 in appendix A.

requirements while having effectively no dispersion at the same time. The last column in figure 5 shows the averaged signals from the control fields. The NN also learned to set all control fields to zero, once the GHZ state has been reached.

6.3. Eigenstate preparation in a quantum parametric oscillator

As the last example, we aim at the preparation of a specific eigenstate of the drift Hamiltonian from equation (9). The initial state is, again, considered as noisy,

$$|\psi(t_0)\rangle = |0\rangle + \sum_{n=0}^{D-1} \mathrm{e}^{-\frac{n}{3}} \xi_n |n\rangle. \tag{13}$$

In our numerical implementation we truncate the Hilbert space of photons to a finite dimension D by considering only the first D Fock states $|n\rangle$. Random noise ξ_n is uniformly sampled from an interval $[-\xi, \xi]$ where ξ is a fixed parameter of the environment. The exponential factor guarantees that the contribution of the highest-lying Fock states is reduced, *i.e.* $|\psi(t_0)\rangle$ is dominantly distributed among the low-lying Fock states.

Let $|\alpha\rangle$ be a coherent state, defined as $a|\alpha\rangle = \alpha|\alpha\rangle$, with annihilation operator a and complex α [38]. These states are often referred to as 'the most classical ones' as they satisfy the minimal Heisenberg uncertainty relation. We consider an example of a Schrödinger cat state $|\mathrm{cat}_\alpha\rangle = \frac{1}{\sqrt{2}}(|\alpha\rangle + |-\alpha\rangle)$ [39]. The respective Wigner quasiprobability distribution is formed by two Gaussians in phase space located at the coordinates ($x = \sqrt{2}\,\mathrm{Re}\alpha, p = \sqrt{2}\,\mathrm{Im}\alpha$) with a non-classical interference pattern between them. Such states can serve as resources for quantum computing, since they can encode a qubit protected against phase-flip errors [40–42]. In our setting of the model, a specific cat state with $\alpha = 2$ happens to be the first excited eigenstate of the drift Hamiltonian. We further refer to this state as the *eigen-cat state*. The goal of this section is to transfer the initial state $|\psi(t_0)\rangle$ to the target $|\psi_{\mathrm{tar}}\rangle = |\mathrm{cat}_2\rangle$.

The loss function $\mathfrak{L}$, again, has the form prescribed by equation (11). Together with other hyperparameters, the coefficients c_μ are presented in table A1 in appendix A. In this case, they were tuned empirically to achieve a desirable performance of the agent[3]. The results are collected in figure 6. As shown in panel (a), after around 300 initial epochs, the agent starts learning quickly. The learning process is stable without any significant oscillations. The performance of the trained NN is shown in panels (b)–(e). The mean fidelity of the test set reaches $\bar{F} \approx 0.93$ with a small variance. The state at the end of the control interval of a randomly chosen example from the test set is depicted in panels (c) and (d). The respective pulse applied is shown in panel (e). The pulse oscillates nearly symmetrically around zero which translates to oscillatory tilting of the classical potential to left and right, c.f. figure 3(c). This qualitative behavior reflects the symmetry of the control task where the target state is equally distributed in both wells and the initial one (though slightly perturbed) at the top of the barrier. As the fidelity approaches its final value, the amplitude of the control field decreases until it only oscillates noisily around zero.

Let us compare to a known protocol to prepare an eigen-cat state; it is based on an adiabatic attenuation of the amplitude G of the parametric drive from equation (9), see reference [43] for recent experimental result. While this is also easily revealed by our agent, we assume here the amplitude G to be a given constant over time and allow the agent only to control the amplitude of an additional single-photon drive. For this problem, the exact pulse shape is not easily anticipated, but our agent managed to find a non-trivial sequence with final fidelities comparable to reference [43].

7. Discussion and outlook

For the systems tested, the differentiable programming method showed promising results and reliably found successful protocols for eigenstate preparation. The optimal strategies can be obtained after a few hundreds of training epochs. The method is intrinsically robust towards uncertainties in the input data. Thus provides more versatility compared to standard quantum control algorithms. Also, the convergence is smooth and stable with respect to different initial seeds and small variations of the hyperparameters.

Despite the uncertainty in the initial states, we managed to reach fidelities larger than 99.9% in the preparation of a GHZ state in a chain of M qubits. We exclusively used fully-connected layers, which limits the depth and thus also the representational power of the NN. Therefore, our current architecture is not well suited for high dimensional (more than 1000) inputs.

To handle even larger input sizes, modifications of the network architecture, which leverage structure assumption on the input wave function, might become necessary. Our current algorithm relies on a classical simulation of a quantum many-body system and is limited by the rapidly growing dimension of the Hilbert space with M. More sophisticated representations of the wave function, such as matrix product states [44] or the multiconfigurational time-dependent Hartree method for indistinguishable particles (MCTDH-X) [45, 46], bear the potential to further improve the scalability of the simulations. Modifications of our method towards classical-quantum hybrid algorithms that incorporate near-term quantum devices are also conceivable [47, 48].

In the case of the quantum parametric oscillator, the eigen-cat state preparation was accomplished with high fidelities despite the noisy input data. In our current setting, the agent has full information about the state at each time step. To better represent experimental reality, some form of photon field measurement could be included. The evolution of such an open system can be described by a stochastic Schrödinger equation (SSE). Optimal control with stochastic dynamics has drawn attention recently, see references [34, 49]. Our method can be applied to open systems in a straightforward way by replacing the current ODE by a SSE.

Furthermore, the robustness of the performance of the agent against experimental imperfections could be investigated within the current scheme. For instance, uncertainties of the pulse parameters could be treated in the same way as the noise in the input states, *i.e.* by adding noise to the respective control amplitudes and sampling over its distribution during the training.

In contrast to the approaches leveraging data from experimental measurements [50], we require a model of the physical system to set up the control tasks. However, recent advantages in parameter estimation from data have the potential to bridge this gap [49, 51]. Furthermore, a student/teacher approach, similar to reference [13], is a conceivable solution.

8. Conclusion

In summary, we have introduced a differentiable programming method for quantum control that leverages information from the gradient obtained by differentiation through the dynamical equations of the system.

[3]Hyperparameter optimization like BOHB could also be employed. However, to create a probabilistic model a relatively large minimum budget is required such that the task is computationally expensive.

Table A1. Hyperparameters employed to obtain the results in the main text. 'LLS 1' stands for first linear layer state-aware network, etc. The numbers of input and output channels of the layers are specified in the brackets.

	parametric osc.	spin chain				qubit	
	eigen-cat	$M = 3$	$M = 4$	$M = 5$	$M = 6$	multiple losses	single loss
Hilbert space							
D	16	8	16	32	64	2	2
ODE solver							
N	187	30	40	50	60	150	150
N_{sub}	200	20	20	20	20	20	20
dt	$10^{-4}U$	0.001 J	0.001 J	0.001 J	0.001 J	0.001 ω	0.001 ω
loss function							
γ	0.999	1.0	1.0	1.0	1.0	1.0	1.0
c_F	0.8	$1.8 \cdot 10^{-4}$	$1.1 \cdot 10^{-4}$	$2.0 \cdot 10^{-4}$	$3.0 \cdot 10^{-4}$	0.57	$3.1 \cdot 10^{-4}$
c_{FN}	200	$8.1 \cdot 10^{-6}$	$3.6 \cdot 10^{-7}$	0.13	0.15	$2.6 \cdot 10^{-3}$	0
c_{amp}	0.01	0	0	0	0	0	0
c'_{amp}	0	$4.2 \cdot 10^{-5}$	$4.1 \cdot 10^{-6}$	$1.7 \cdot 10^{-6}$	$1.7 \cdot 10^{-6}$	$3.9 \cdot 10^{-6}$	0
Adam							
learning rate	$4 \cdot 10^{-5}$	$8.4 \cdot 10^{-4}$	$7.0 \cdot 10^{-4}$	$6.0 \cdot 10^{-4}$	$6.0 \cdot 10^{-4}$	$3.3 \cdot 10^{-3}$	$4.9 \cdot 10^{-4}$
b	64	256	512	256	256	256	256
epochs	3000	2000	2000	3000	4000	400	400
NN							
LLS 1	(32, 512)	(16, 512)	(32, 256)	(64, 128)	(128, 256)	(4, 256)	(4, 256)
LLS 2	(512, 256)	(512, 32)	(256, 256)	(128, 128)	(256, 256)	(256, 256)	(256, 256)
LLS 3	(256, 256)			(128, 128)	(256, 256)	(256, 128)	(256, 128)
LLS 4	(256, 64)						
LLA 1	(1, 128)	(6, 16)	(8, 64)	(10, 32)	(12, 64)	(1, 128)	(1, 128)
LLA 2	(128, 64)	(16, 32)	(64, 256)	(32, 32)	(64, 64)	(128, 128)	(128, 128)
LLA 3				(32, 128)	(64, 256)		
LLC 1	(64, 64)	(32, 32)	(256, 256)	(128, 128)	(256, 256)	(128, 64)	(128, 64)
LLC 2	(64, 32)	(32, 6)	(256, 8)	(128, 128)	(256, 256)	(64, 32)	(64, 32)
LLC 3	(32, 2)			(128, 10)	(256, 12)	(32, 1)	(32, 1)

The approach was successfully demonstrated on a single- and a many-body quantum system. The method is intrinsically robust towards uncertainty in the input states. Moreover, its application to open quantum systems is straightforward. Due to these attributes, we hope it will become a useful part of physicists' toolbox to control complex quantum systems.

Acknowledgment

We would like to thank Eliska Greplova and Martin Koppenhöfer for inspiring discussions. This work was financially supported by the Swiss National Science Foundation (SNSF) and the NCCR Quantum Science and Technology. Calculations were performed at sciCORE (scicore.unibas.ch) scientific computing core facility at University of Basel.

Data availability statement

The codes that support the findings of this study are openly available [52].

Appendix A. Hyperparameters of the models

The hyperparameters used in the control tasks are summarized in table A1.

Appendix B. Hyperparameter optimization

In this appendix, we illustrate the results obtained by the hyperparameter optimization with the Bayesian Optimization and Hyperband method (BOHB) [32] in the case of the qubit control task with $\mathfrak{L} = c_F \mathfrak{L}_F$ (see

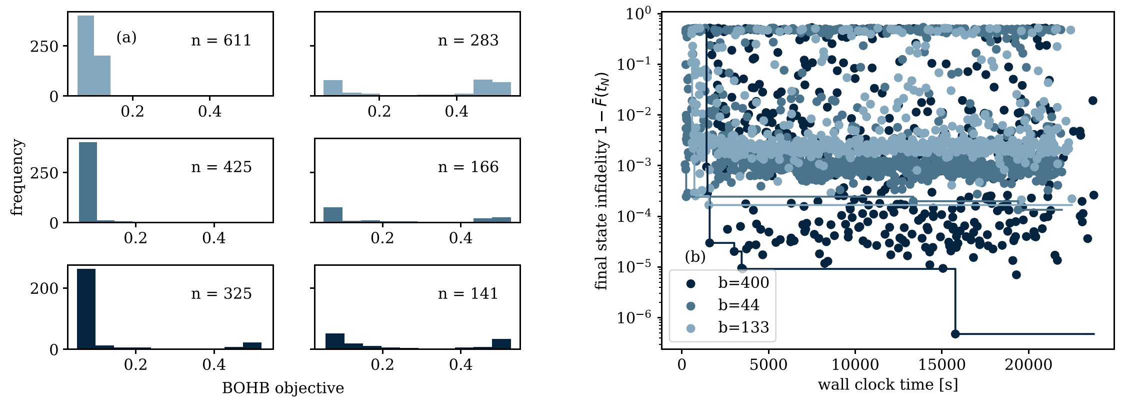


Figure B1. BOHB optimization of the hyperparameters for the manipulation of a single qubit (8) with figure of merit chosen as $\mathfrak{L} = c_F\mathfrak{L}_F$ (see equation (11)). Panel (a): Frequency distribution as a function of the BOHB objective for model-based picks (left) versus random-configurations (right) for three budgets (rows). The number n in each cell indicates the total number of samples that fall into the category. Panel (b): Final state infidelity, $1-\bar{F}(t_N)$, as a function of the duration of the simulation (total wall time). The budgets—and data—are identical to (a), as highlighted by the color code.

equation (11)). BOHB is based on the well-known Hyperband approach [53] to determine how many hyperparameter configurations are evaluated at a certain budget, c.f. pseudocode in reference [32]. In our case, the budget is identical to the number of epochs that are used to train the predictive model. In contrast to Hyperband, BOHB replaces the random selection of configurations at the beginning of each iteration by a model-based search. Figure B1(a) visualizes the frequency distribution as a function of the BOHB objective, $\mathfrak{L}_F$, for model-based and random configurations. As desired, nearly all model-based picks achieve a very small value for $\mathfrak{L}_F$. For both distributions, an increase of the budget leads to smaller values of $\mathfrak{L}_F$ on average, as expected. The differences between the three budgets can also be seen in the evolution of the final state infidelity, $1-\bar{F}(t_N)$, as a function of the wall clock time in figure B1(b).

The wall clock time refers to the actual run time of the optimization or, in other words, it measures the elapsed time between the start of the optimization and the end of a given task in the BOHB iterations. We stress that this time strongly depends on the hardware used for the simulation. Please note that the infidelity is plotted in log-scale and that many configurations reach a very high fidelity.

Appendix C. Differentiable programming versus REINFORCE

Here we compare the performance of our differentiable programming (DP) method with a vanilla policy gradient algorithm (REINFORCE) [9, 16, 54, 55] in the case of the qubit control problem from section 6.1. In a nutshell, our REINFORCE implementation is based on three substeps: Firstly, trajectories τ are sampled from the current Gaussian policy

$$\log\pi_\theta(u_x(t_i)|\{|\psi(t_i)\rangle, u_x(t_{i-1})\}) = -\frac{1}{2}\left(\frac{(u_x(t_i)-\mu)^2}{\sigma^2} + \log(2\pi\sigma^2)\right),$$

where the variance $\sigma^2 = 0.04$ is fixed and the mean $\mu = \mu_\theta(\{|\psi(t_i)\rangle, u_x(t_{i-1})\})$ is determined by the predictive model, which is structured identically to the one in the DP case, see table A1 in appendix A. Secondly, based on the rewards-to-go

$$R_n = \sum_{n'=n}^{N}\left[c_F|\langle\psi(t_{n'})|\psi_{\rm tar}\rangle|^2 - c'_{\rm amp}|u_x(t_n)|^2 + \delta_{n',N}c_{FN}|\langle\psi(t_{n'})|\psi_{\rm tar}\rangle|^2\right],$$

where the (properly adjusted) elementary loss functions from section 5 are employed, we approximate the gradient of the expected total reward over all trajectories J_θ as

$$\nabla_\theta J_\theta = \mathop{\mathrm{E}}_{\tau\sim\pi_\theta}\left[\sum_{n=0}^{N}\nabla_\theta\log\pi_\theta(u_x(t_n)|\{|\psi(t_n)\rangle, u_x(t_{n-1})\})R_n\right].$$

Note that while the loss functions for the DP are minimized, the rewards in RL are maximized. Note further that the rewards-to-go are normalized before they enter the approximation of the gradient [56]. Thirdly, we use the ADAM optimizer with a learning rate of 0.000 25 to update the network parameters in a series of epochs, each consisting of 1024 trajectories.

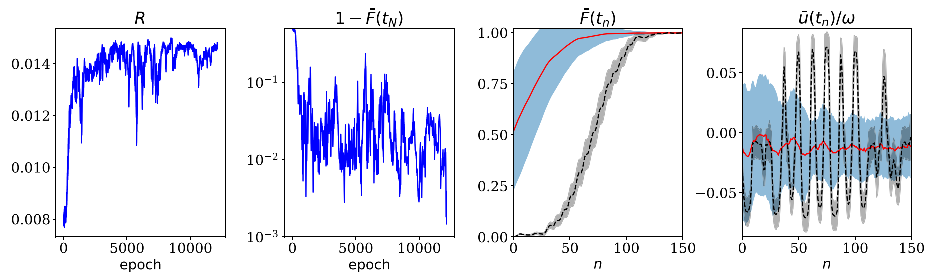


Figure C2. Preparation of the eigenstate $|\uparrow\rangle_z$ of the drift term of the Hamiltonian (8) for the optimized hyperparameters containing the coefficients c_F, c_{FN}, c'_{amp} based on a REINFORCE implementation. In the first column, the evolution of the mean reward R during the training phase is shown. The second column shows the mean value of the final state infidelity as a function of the epochs. The third and fourth columns show the performance of the model when it is applied to a test set of size 512. The red curve/ blue shaded region indicate the mean/standard deviation of the fidelity and the actions, respectively. The black dashed line highlights the special case with initial state given as $|\psi(t_0)\rangle = |\downarrow\rangle_z$.

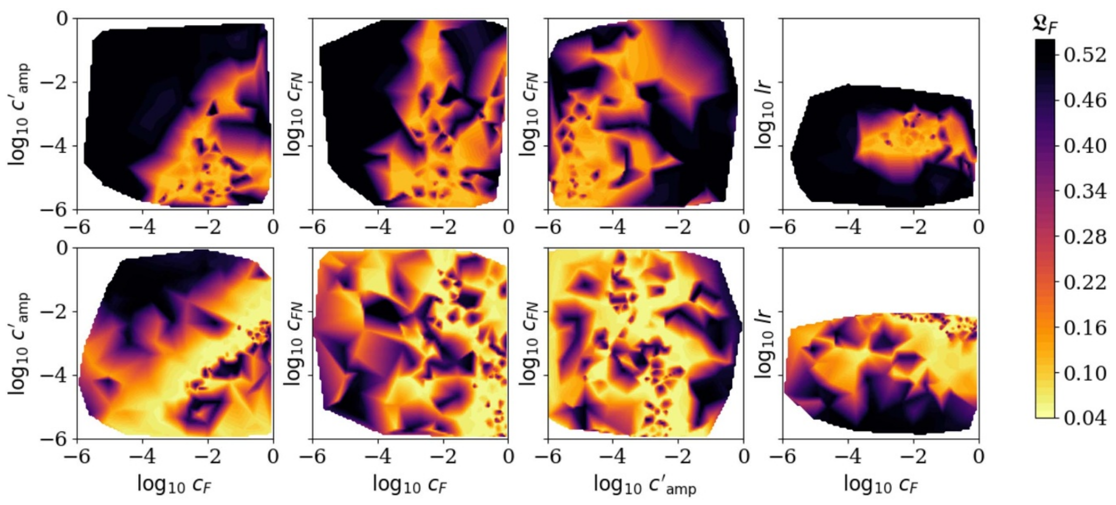


Figure C3. Optimal regions in the hyperparameter space consisting of c_F, c'_{amp}, c_{FN} and learning rate lr of ADAM as established by BOHB for the DP approach (bottom row) and the RL approach (top row). The figures show projections from the four-dimensional space onto the respective two-dimensional subspaces. The color map encodes the main objective of BOHB to be minimized, *i.e.* the average infidelity over the entire time interval $\mathfrak{L}_F$. The brighter areas correspond to a better performance. The continuous map results from an interpolation between the discrete set of sampled configurations probed by the BOHB algorithm.

Figure C2 is analogous to figure 4 with RL replacing DP. In general, we obtain a noisier evolution of the total expected reward during training as compared to the (relatively) smooth appearance of the training loss in the main text. Furthermore, the outcome is more sensitive to small changes of hyperparameters including the employed random seed. We expect that more sophisticated approaches, such as trust-region policy optimization [57] or proximal policy optimization [58], in combination with generalized advantage estimation [59] could improve the performance. As expected, more epochs were needed to train the neural network. Also, the agent did not learn to push the control fields towards zero once the target state has been prepared.

A comparison between the mean infidelity $\mathfrak{L}_F$ over all time steps as a function of the hyperparameters c_F, c_{FN}, c'_{amp} and the learning rate lr for both approaches is depicted in figure C3. What is plotted are the two-dimensional projections of $\mathfrak{L}_F$ onto the subspaces of the respective pairs of hyperparameters. The fact that the landscapes for the DP method (lower panel) reach lower infidelities as compared to the REINFORCE implementation (top panel) shows that the DP method generally performs better. Also the bright regions, indicating a successful performance, are more extended in our approach which implies that it is more stable with respect to changes in the hyperparameters. Remarkably, DP is trained only for 400 epochs while 10 000 epochs are used in case of the REINFORCE algorithm.

ORCID iDs

Frank Schäfer https://orcid.org/0000-0003-2684-4984
Michal Kloc https://orcid.org/0000-0002-4575-7723

References

[1] Dong D and Petersen I R 2010 Quantum control theory and applications: a survey *IET control theory & applications* **4** 2651–71
[2] Glaser S J *et al* 2015 Training Schrödinger's cat: quantum optimal control *Eur. Phys. J.*D **69** 279
[3] Krotov V F 1989 Global methods to improve control and optimal control of resonance interaction of light and matter *Modeling and Control of Systems* ed Blaquiére A (Berlin, Heidelberg: Springer Berlin Heidelberg) pp 267–98
[4] Khaneja N, Reiss T, Kehlet C, Schulte-Herbrüggen T and Glaser S J 2005 Optimal control of coupled spin dynamics: design of NMR pulse sequences by gradient ascent algorithms *J. Magn. Reson.* **172** 296–305
[5] Chen C, Dong D, Long R, Petersen I R and Rabitz H A 2014 Sampling-based learning control of inhomogeneous quantum ensembles *Phys. Rev.*A **89** 023402
[6] Chen Q M, Wu R B, Zhang T M and Rabitz H 2015 Near-time-optimal control for quantum systems *Phys. Rev.*A **92** 063415
[7] Morzhin O V and Pechen A N 2019 Krotov method for optimal control of closed quantum systems *Russian Mathematical Surveys* **74** 851
[8] Leung N, Abdelhafez M, Koch J and Schuster D 2017 Speedup for quantum optimal control from automatic differentiation based on graphics processing units *Phys. Rev.*A **95** 042318
[9] Sutton R S and Barto A G 2018 *Reinforcement Learning: An Introduction* 2nd ed (Cambridge, MA: MIT Press) (http://incompleteideas.net/book/the-book-2nd.html)
[10] Bukov M, Day A G R, Sels D, Weinberg P, Polkovnikov A and Mehta P 2018 Reinforcement learning in different phases of quantum control *Phys. Rev.* X **8** 031086
[11] Bukov M 2018 Reinforcement learning for autonomous preparation of Floquet-engineered states: Inverting the quantum Kapitza oscillator *Phys. Rev.*B **98** 224305
[12] Niu M Y, Boixo S, Smelyanskiy V N and Neven H 2019 Universal quantum control through deep reinforcement learning *npj Quantum Information* **5** 33
[13] Fösel T, Tighineanu P, Weiss T and Marquardt F 2018 Reinforcement learning with neural networks for quantum feedback *Phys. Rev.* **8** 031084
[14] Atkeson C G and Santamaria J C 1997 A comparison of direct and model-based reinforcement learning *Proc. of Int. Conf. on Robotics and Automation* (IEEE) vol 4 pp 3557–64 (https://ieeexplore.ieee.org/abstract/document/606886)
[15] Kurutach T, Clavera I, Duan Y, Tamar A and Abbeel P 2018 *Model-Ensemble Trust-Region Policy Optimization* (arXiv:1802.10592)
[16] Achiam J 2018 *Spinning up in deep RL* (https://spinningup.openai.com/en/latest/)
[17] Baydin A G, Pearlmutter B A, Radul A A and Siskind J M 2018 Automatic differentiation in machine learning: a survey *J. Machine Learning Research* **18** 1–43 (http://jmlr.org/papers/v18/17-468.html)
[18] Innes M, Neethu M J and Karmali T 2019 *Reinforcement Learning vs. Differentiable Programming* (https://fluxml.ai/2019/03/05/dp-vs-rl.html)
[19] Chen T Q, Rubanova Y, Bettencourt J and Duvenaud D K 2018 Neural ordinary differential equations *Advances in Neural Information Processing Systems 31 (NIPS 2018)* pp 6571–83 (http://papers.nips.cc/paper/7892-neural-ordinary-differential-equations)
[20] Degrave J, Hermans M, Dambre J and Wyffels F 2019 A differentiable physics engine for deep learning in robotics *Front. Neurobiot.* **13** 6
[21] de Avila Belbute-Peres F, Smith K, Allen K, Tenenbaum J and Kolter J Z 2018 End-to-end differentiable physics for learning and control *Advances in Neural Information Processing Systems 31 (NIPS 2018)* pp 7178–89 (http://papers.nips.cc/paper/7948-end-to-end-differentiable-physics-for-learning-and-control.pdf)
[22] Innes M *et al* 2019 *Zygote: A differentiable programming system to bridge machine learning and scientific computing* (arXiv:1907.07587)
[23] Rackauckas C, Ma Y, Martensen J, Warner C, Zubov K, Supekar R, Skinner D and Ramadhan A 2020 *Universal Differential Equations for Scientific Machine Learning* (arXiv: 2001.04385)
[24] Liao H J, Liu J G, Wang L and Xiang T 2019 Differentiable programming tensor networks *Phys. Rev.* X **9** 031041
[25] Doria P, Calarco T and Montangero S 2011 Optimal control technique for many-body quantum dynamics *Phys. Rev. Lett.* **106** 190501
[26] Caneva T, Calarco T and Montangero S 2011 Chopped random-basis quantum optimization *Phys. Rev.*A **84** 022326
[27] Wu R B, Chu B, Owens D H and Rabitz H 2018 Data-driven gradient algorithm for high-precision quantum control *Phys. Rev.*A **97** 042122
[28] Wu R B, Ding H, Dong D and Wang X 2019 Learning robust and high-precision quantum controls *Phys. Rev.*A **99** 042327
[29] Paszke A *et al* 2017 Automatic differentiation in PyTorch *Nips-W* https://openreview.net/forum?id=BJJsrmfCZ
[30] Süli E and Mayers D F 2003 *An Introduction to Numerical Analysis* (Cambridge: Cambridge University Press)
[31] Kingma D P and Ba J 2014 *Adam: A Method for Stochastic Optimization* (arXiv:1412.6980)
[32] Falkner S, Klein A and Hutter F 2018 *BOHB: Robust and efficient hyperparameter optimization at scale* (arXiv:1807.01774)
[33] Bartolo N, Minganti F, Casteels W and Ciuti C 2016 Exact steady state of a Kerr resonator with one-and two-photon driving and dissipation: Controllable Wigner-function multimodality and dissipative phase transitions *Phys. Rev.*A **94** 033841
[34] Abdelhafez M, Schuster D I and Koch J 2019 Gradient-based optimal control of open quantum systems using quantum trajectories and automatic differentiation *Phys. Rev.*A **99** 052327
[35] Greenberger D M, Horne M A and Zeilinger A 1989 Going beyond Bell's theorem *Bell's Theorem, Quantum Theory and Conceptions of the Universe* (Berlin: Springer) pp 69–72
[36] Bose S, Vedral V and Knight P L 1998 Multiparticle generalization of entanglement swapping *Phys. Rev.*A **57** 822–9
[37] Hillery M, Bužek V and Berthiaume A 1999 Quantum secret sharing *Phys. Rev.*A **59** 1829–34
[38] Glauber R J 1963 Coherent and incoherent states of the radiation field *Phys. Rev.* **131** 2766–88
[39] Leibfried D *et al* 2005 Creation of a six-atom 'Schrödinger cat' state *Nature* **438** 639
[40] Cochrane P T, Milburn G J and Munro W J 1999 Macroscopically distinct quantum-superposition states as a bosonic code for amplitude damping *Phys. Rev.*A **59** 2631–4
[41] Mirrahimi M *et al* 2014 Dynamically protected cat-qubits: a new paradigm for universal quantum computation *New J. Phys.* **16** 045014
[42] Grimm A *et al* 2019 *The Kerr-cat Qubit: Stabilization, Readout and Gates* (arXiv:1907.12131)
[43] Wang Z *et al* 2019 Quantum dynamics of a few-photon parametric oscillator *Phys. Rev.* X **9** 021049
[44] Wall M L and Carr L D 2012 Out-of-equilibrium dynamics with matrix product states *New J. Phys.* **14** 125015

[45] Lode A U J, Lévêque C, Madsen L B, Streltsov A I and Alon O E 2020 Colloquium: Multiconffgurational time-dependent Hartree approaches for indistinguishable particles *Rev. Mod. Phys.* **92** 011001
[46] Schäfer F, Bastarrachea-Magnani M A, Lode A U J, de Forges de Parny L and Buchleitner A 2020 Spectral structure and many-body dynamics of ultracold bosons in a double-well *Entropy* **22** 382
[47] Mitarai K, Negoro M, Kitagawa M and Fujii K 2018 Quantum circuit learning *Phys. Rev.*A **98** 032309
[48] Benedetti M, Lloyd E, Sack S and Fiorentini M 2019 Parameterized quantum circuits as machine learning models *Quantum Science and Technology* **4** 043001
[49] Krastanov S, Zhou S, Flammia S T and Jiang L 2019 Stochastic estimation of dynamical variables *Quantum Science and Technology* **4** 035003
[50] Ferrie C and Moussa O 2015 Robust and efficient in situ quantum control *Phys. Rev.*A **91** 052306
[51] Flurin E, Martin L S, Hacohen-Gourgy S and Siddiqi I 2020 Using a recurrent neural network to reconstruct quantum dynamics of a superconducting qubit from physical observations *Phys. Rev.* X **10** 011006
[52] Schäfer F, Kloc M, Bruder C and Lörch N 2020 *A Differentiable Programming Method for Quantum Control* (https://github.com/frankschae/A-differentiable-programming-method-for-quantum-control)
[53] Li L, Jamieson K, DeSalvo G, Rostamizadeh A and Talwalkar A 2017 Hyperband: A novel bandit based approach to hyperparameter optimization *The Journal of Machine Learning Research* **18** 6765–816
[54] Williams R J 1992 Simple statistical gradient-following algorithms for connectionist reinforcement learning *Mach. Learn.* **8** 229–56
[55] Sutton R S, McAllester D A, Singh S P and Mansour Y 2000 Policy gradient methods for reinforcement learning with function approximation *Advances in Neural Information Processing Systems 55* 1057–63 (http://papers.nips.cc/paper/1713-policy-gradient-methods-for-reinforcement-learning-with-function-approximation.pdf)
[56] Karpathy A 2016 Deep reinforcement learning: Pong from pixels (http://karpathy.github.io/2016/05/31/rl/)
[57] Schulman J, Levine S, Abbeel P, Jordan M and Moritz P 2015 Trust region policy optimization *Int. Conference on Machine Learning* pp 1889–97 (http://proceedings.mlr.press/v37/schulman15.pdf)
[58] Schulman J, Wolski F, Dhariwal P, Radford A and Klimov O 2017 *Proximal Policy Optimization Algorithms* (arXiv:1707.06347)
[59] Schulman J, Moritz P, Levine S, Jordan M and Abbeel P 2015 *High-Dimensional Continuous Control Using Generalized Advantage Estimation* (arXiv:1506.02438)