When reinforcement learning stands out in quantum control? A comparative study on state preparation

Xiao-Ming Zhang,^{1,2} Zezhu Wei,¹ Raza Asad,³ Xu-Chen Yang,⁴ and Xin Wang^{1,2}

¹Department of Physics, City University of Hong Kong,

Tat Chee Avenue, Kowloon, Hong Kong SAR, China

²City University of Hong Kong Shenzhen Research Institute, Shenzhen, Guangdong 518057, China

³Department of Mathematics, City University of Hong Kong,

Tat Chee Avenue, Kowloon, Hong Kong SAR, China

⁴Department of Physics, The University of Hong Kong, Pokfulam, Hong Kong SAR, China

Reinforcement learning has been widely used in many problems including quantum control of qubits. However, such problems can, at the same time, be solved by traditional, non-machine-learning based methods such as stochastic gradient descendent and Krotov algorithms, and it remains unclear which one is most suitable when the control has specific constraints. In this work we perform a comparative study on the efficacy of two reinforcement learning algorithms, Q-learning and deep Q-learning, as well as stochastic gradient descendent and Krotov algorithms, in the problem of preparing a desired quantum state. We found that overall, the deep Q-learning outperforms others when the problem is discretized, e.g. allowing discrete values of control. The Q-learning and deep Q-learning can also adaptively reduce the complexity of the control sequence, shortening the operation time and improving the fidelity. Our comparison provides insights on the suitability of reinforcement learning in quantum control problems.

Introduction

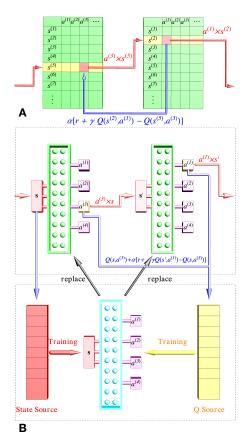
Reinforcement learning, a branch of machine learning in artificial intelligence, has proven to be a powerful tool to solve a wide range of complex problems, such as the games of Go [1] and Atari [2]. Reinforcement learning has also been applied to a variety of problems in quantum physics with vast success [3–15], including the quantum state preparation [3-6], state transfer [7], quantum gate design [8], and error correction [9]. In many cases, it outperforms commonly-used conventional algorithms, such as Krotov and Stochastic Gradient Descendent (SGD) algorithms [7, 8]. In the reinforcement learning algorithm, an optimization problem is converted to a set of policies that governs the behavior of a computer agent, i.e. its choices of actions and, consequently, the reward it receives. By simulating sequences of actions taken by the agent maximizing the reward, one finds an optimal solution to the desired problem [16].

While there is currently a frenetic attempt to apply reinforcement learning and other machine-learning-based algorithms [17–22] to a wide range of physics problems, a fundamental question arises: under what situation the reinforcement learning is the most suitable method? As examples, we consider problems related to quantum control of a qubit. The goal of these problems is typically to steer the qubit toward a target state under certain constraints, and the mismatch between the final qubit state and the target state naturally serves as both the cost function used in the SGD or Krotov methods, and the reward that can be applied in the reinforcement learning procedure. Our question then becomes: under different scenarios of constraints, which algorithm is the best? In this work, we compare the efficacy two commonly-used traditional

methods: SGD and the Krotov method, and two algorithms based on reinforcement learning: Q-learning (QL) and deep Q-learning (deep QL or DQL), under situations with different types of control constraints.

In [3], the Q-learning technique has been applied to the problem of quantum state preparation, revealing interesting physics of different stages of quantum control. The problem of preparing a desired quantum state from a given initial state is on one hand simple enough to be investigated in full detail, and on the other hand containing sufficient physics allowing for various types of control constraints. We therefore take quantum state preparation as the platform that our comparison of different algorithms are based on. While detailed description of the quantum state preparation is provided in Results, we briefly introduce the four algorithms we are comparing in this work here. (Detailed implementations are provided in the Supplemental Material.)

SGD and Krotov. SGD is one of the simplest gradient-based optimization algorithms. In each iteration, a direction in the parameter space is randomly chosen, along which the control field is updated using the gradient of the cost function defined as the mismatch between the evolved state and the target state. Ideally the gradient is zero when the calculation has converged to the optimal solution. The Krotov algorithm [23] has a different strategy: The initial state is first propagated forward obtaining the evolved state. The evolved state is then projected to the target state, defining a co-state encapsulating the mismatch between the two. Then the co-state is propagated backward to the initial state, during which process the control fields are updated. When the calculation is converged, the co-state is identical to the target state.



Sketch of the procedure of Q-learning and deep Q-learning algorithms. (A) In Q-learning, the Q(s,a) values are stored in the Q-table. When the agent is at state $s^{(5)}$, it reviews $Q(s^{(5)}, a^{(i)})$ for all possible actions and chooses one with the maximum probability, or "Q-value" (which we assume is $a^{(3)}$). As a result, the state then evolves to $s^{(2)}$. Depending on the distance between $s^{(2)}$ and the target, the Q-values (e.g. $Q(s^{(5)}, a^{(3)})$) is updated according to Eq. (7). This process is then repeated at the new state $s^{(2)}$ and so forth. (B) In deep Q-learning, the Q-table is replaced by the Q-network. Instead of choosing an action with the maximum Q-value from a list, this process is done by a neural network, the Q-network, which takes the input state (s)and outputs an action that it finds most appropriate. Evaluation of the resulting state (s') after the action suggests how the neural network should be updated (trained). For detailed implementation, see Methods and Supplemental Material.

QL and deep **QL**. In reinforcement learning, a computer agent is allowed to be in a set of states, and in each step the agent chooses an action bringing it to another state. By performing a set of actions, the agent acquires a reward, which encapsulates the ingredients of the optimization problem that one desires to solve. Fig. 1A schematically shows how QL works. At each state s, the agent has a choice of a set of actions a each having certain probability called the action-value function Q(s,a), forming the so-called Q-table. The agent typically chooses the action with the largest probability, and by comparing the

resulting state to the target state, a reward is fed back to update the Q-table so that actions that lead closer to the target will have larger probability to be chosen. By iterating this process sufficient times, one finds an optimal solution. We note that since the table should have a finite number of entries, both the states and actions should be discretized. Fig. 1B shows deep QL, in which the role of the Q-table is replaced by a neural network, called the Q-network. The agent then chooses its action according to the output of the Q-network, and the reward is used to update the network. In this case, although the allowed actions must still be discrete (the network has a finite number of neurons in the output layer), the input state can actually be continuous. Moreover, as we shall see in this paper, the Q-network is much more powerful than the Q-table, making deep QL more efficient.

Results

We consider the time dependent Hamiltonian

$$H[J(t)] = 4J(t)\sigma_z + h\sigma_x,\tag{1}$$

where σ_x and σ_z are Pauli matrices. The Hamiltonian may describe a singlet-triplet qubit [24] or a single spin with energy gap h under tunable control fields [25, 26]. In these systems, it is difficult to vary h during gate operations, and we therefore assume that h is a constant in our work, which at the same time serves as our energy unit. Quantum control of the qubit is then achieved by altering J(t) dynamically.

Quantum state preparation refers to the problem to find J(t) such that a given initial state $|\psi_0\rangle$ evolves, within time T, to a final state $|\psi_f\rangle$ that is as close as possible to the target state $|\phi\rangle$. The quality of the state transfer is evaluated using the fidelity, defined as

$$F = \left| \langle \psi_f | \phi \rangle \right|^2, \tag{2}$$

and we typically use the averaged fidelity \overline{F} over many runs of a given algorithm in our comparison (unless otherwise noted, we average 100 runs to obtain \overline{F}).

In this work, we take $|\psi_0\rangle = |0\rangle$, $|\phi\rangle = |1\rangle$ and $T = 2\pi$ unless otherwise specified. Under different situations, there are various kinds of preferences or restrictions of control. We consider the following types of restrictions:

(i) Assuming that control is performed with a sequence of piecewise constant pulses, and in this work we further assume that the time duration of each piece is equal to each other for convenience. For this purpose, we divide the total time T into N equal time steps, each of which having a step size dt = T/N, with N denoting the maximum number of pieces required by the control. J(t) is accordingly discretized, so that on the ith time step, $J(t) = J_i$ and the system evolves under $H(J_i)$. Denot-

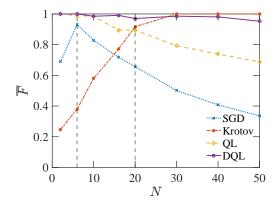


FIG. 2: Average fidelities as functions of the maximum number of control pieces. For QL and deep QL, $J_i \in \{0,1\}$ (i.e. M=1). For SGD and Krotov, no restriction is imposed on the range of J_i and M (i.e. $M \to \infty$). The vertical dashed lines correspond to results shown with respective M values in Figs. 5 and 6.

ing the state at the end of the *i*th time step as $|\psi_i\rangle$, the evolution at the *i*th step is $|\psi_i\rangle = U_i|\psi_{i-1}\rangle$, where $U_i = \exp\{-iH(J_i)dt\}$. In principle, the evolution time can be less than T, namely the evolution may conclude at the i_f th time step with $i_f \leq N$. Due to their nature, SGD and Krotov have to finish all time steps, i.e. $i_f = N$. However as we shall see below, QL and deep QL frequently have $i_f < N$.

(ii) We also consider the case where the magnitude of the control field is bounded, i.e. $J_i \in [J_{\max}, J_{\min}]$ for all i. The constraint can be straightforwardly satisfied in QL and deep QL, since they only operate within the given set of actions thus cannot exceed the bounds. For SGD and Krotov, updates to the control fields may exceed the bounds, in which case we need to enforce the bounds by setting J_i as J_{\max} when the updated value is greater than J_{\min} , and as J_{\min} when the updated value is smaller than J_{\min} . In the case in which either of them is not restricted, we simply note $J_{\min} \to -\infty$ or $J_{\max} \to \infty$.

(iii) The values of the control field may be discretized in the given range, i.e., $J_i \in \{J_{\min}, J_{\min} + \mathrm{d}J/M, J_{\min} + 2\mathrm{d}J/M, \cdots, J_{\max}\}$ where $\mathrm{d}J = (J_{\max} - J_{\min})/M$, so that the control field can take M+1 values including J_{\min} and J_{\max} . In reality this situation may arise, for example, when decomposing a quantum operation into a set of given gates [27–29]. For a reason similar to (ii), QL and deep QL only select actions within the given set so the constraint is satisfied. For SGD and Krotov which keep updating the values of the control field during iterations, we enforce the constraint by setting the value of each control field to the nearest allowed value at the end of the execution.

To sum up, the number of pieces in control sequences N, the bounds of the control field J_{\min} and J_{\max} , as well as the number of the discrete values of the control field M+1 are the main factors characterizing situations to

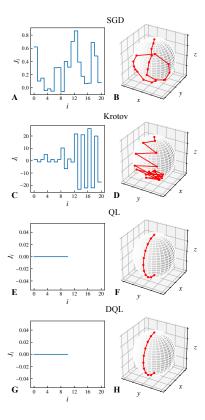


FIG. 3: Pulse profiles and the corresponding trajectories on Bloch sphere. Left column: Example pulse profiles taken from results of Fig. 2 with N=20. Right column: Evolution of the state corresponding to the respective control sequence on the left column.

prepare quantum states, based on which our comparison of different algorithms is conducted. We also define $N^{\rm iter}$ as the number of iterations performed in executing an algorithm, which is typically taken as equal for different algorithms to ensure a fair comparison. Unless otherwise noted, $N^{\rm iter}=500$ in all results shown. In this work, the quality of an algorithm is assessed by the averaged fidelity of the state it prepares (as compared to the target state) \overline{F} , but not by the computational resources it costs.

In Fig. 2 we study a situation where the maximum number of pieces in the control sequence N is given, and the results are shown as the averaged fidelities as functions of N. For $N \leq 10$, the Krotov method gives the lowest fidelity, possibly due to the fact that Krotov requires a reasonable level of continuity in the control sequence, and one with a few pieces is unlikely to reach convergence. As N increases, the performance of Krotov is much improved, which has the highest fidelity when N is large ($N \geq 30$ as seen in the figure). SGD performs better than Krotov for N > 10, but worse otherwise, because as N increase, the algorithm has to search over a much larger parameter space. Within the given number of iterations ($N^{\text{iter}} = 500$ as noted above) it concludes with a lower fidelity. Of course, this result can be im-

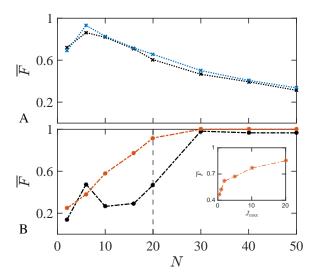


FIG. 4: Effect of bounds of the control on the average fidelities as functions of N for SGD and Krotov methods. (A) \overline{F} versus N for the SGD method, without (blue) and with (black) restriction of $J_i \in [0,1]$. (B) Main panel: \overline{F} versus N for the Krotov method, without (red) and with (black) restriction of $J_i \in [0,1]$. Inset: \overline{F} versus J_{max} for N=20, where J_i is restricted to $J_i \in [1-J_{\text{max}},J_{\text{max}}]$.

proved if more iterations are allowed, and we shall show relevant results in Sec. II of the Supplemental Material. The SGD results at N=2 is irregular (thus the cusp at N=6), due to the lack of flexibility in the control sequence which makes it difficult to achieve high fidelity with only two steps.

The result for QL has higher fidelity than SGD and Krotov (but still lower than that of deep QL), indicates the superior ability of reinforcement learning. Nevertheless, we note that the QL may sometimes fail: it occasionally arrives at a final state which is completely different than the target state. On the other hand, SGD could fail by being trapped at a local minimum, but even in that case it is not drastically different from the optimal solution in terms of the fidelity. This is the reason why the QL results drop for N > 10. For larger N, the failure rate for QL is higher (possibly due to the higher dimensionality of the Q-table), and therefore the averaged fidelity is lower. Among all four algorithms, deep QL is perhaps the best: it gives the highest fidelity for N < 30. For N > 30, the deep QL is outperformed by Krotov, which arises from the nonzero failure probability, but the effect is moderate and the fidelity is still very close to 1. Our results indicate that the deep QL is the most suitable algorithm for this scenario.

To further understand the results shown in Fig. 2, we take examples from N=20 and plot the pulse profiles and the corresponding trajectories on the Bloch sphere in Fig. 3. We immediately realize that the QL and deep QL yield very simple pulse shapes: one only has to keep

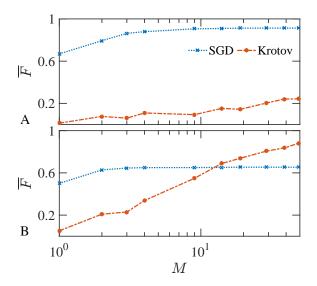


FIG. 5: Effect of discrete control fields on the averaged fidelity for SGD and Krotov methods. In the calculation, the strength of control field is not specifically restricted, so the J_{\min} and J_{\max} values are determined after the algorithm has run. The values of the control field is then mapped to their respective closest discrete values, with the total number of allowed values including J_{\min} and J_{\max} being M+1. Panel (A) shows the case of N=6 while (B) N=20, corresponding to the two vertical dashed lines in Fig. 2.

the control at zero for time T/2, and the desired target state $(|1\rangle)$ will be achieved. However, to find the result, the algorithm has to somehow realize that one does not have to complete all N pieces, which implies their ability to adaptively generating the control sequence. As can be seen from Fig. 3A and 3C, SGD and Krotov only search for pulse sequences with exactly N pieces and therefore miss the optimal solution. Their trajectories on the Bloch sphere are much more complex as compared to those of QL and deep QL. In practice, the complex pulse shapes and longer gate times mean that they are difficult to realize in the laboratory, and potentially introduces error to the control procedure (In Sec. IV of the Supplemental Material we provide more details on this issue). From Fig. 3 we also notice that QL and deep QL possess better ability to adaptively sequencing, which is particularly suitable for problems that involve optimization of gate time or speed, such as the quantum speed limit [3, 7]. On the other hand, application of SGD or Krotov to the same problem requires searching over various different Nvalues before an optimal solution can be found, which cost much more resources [30, 31].

For results shown in Fig. 2, the control field is bounded for QL and deep QL due to their nature that there are a limited set of actions to choose from. However, results for SGD and Krotov are obtained with no restriction imposed on the control field. Here, we discuss the situation that the control field is bounded for SGD and Krotov,

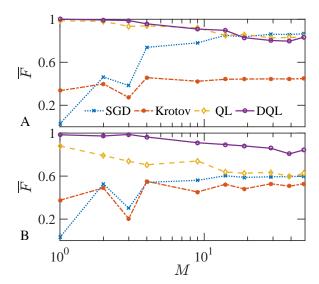


FIG. 6: Effect of discrete control fields on the averaged fidelity for all four methods considered. The strength of control field is restricted to $J_i \in [0, 1]$, and M + 1 discrete values (including 0 and 1) are allowed. Panel (A) shows the case of N = 6 while (B) N = 20, corresponding to the two vertical dashed lines in Fig. 2.

and the results are shown in Fig. 4. Fig. 4A shows the results for the SGD method, with the blue line identical to that in Fig. 2 (no restriction) and the black one showing results after J is restricted between 0 and 1. We see that imposing a restriction on the available range of the control field does not change the results much, because the search by the SGD algorithm is essentially local: the alteration of J is small in each step and it is unlikely to build up a significant variation of J in the final results. This fact can also be seen from Fig. 3A: the strength of control field is mostly within the range of [0,1] so that the restriction has minimal effect on the results.

The situation is different for the Krotov method. As can be seen in Fig. 4B, for N < 30, the result from the Krotov method with restriction of $J_i \in [0,1]$ (black line) has considerably lower average fidelities than that without restriction (red line, identical to the results shown in Fig. 2). This is because the Krotov method makes large updates on the values of the control fields, as can be seen from Fig. 3A where the magnitude of J_i can be above 20. Restricting the control field to a much narrower range will severely compromise the ability of the algorithm to find solutions with high fidelities. An exception is N=6, for which the results with restriction has higher average fidelity. We are uncertain on its cause and believe that the algorithm succeeds in this particular case but not in general. After all, the averaged fidelity is below 0.6 for both lines, with or without restrictions. For N > 30, the results without restriction on the range of the control approaches almost one, and those with restriction is lower than one but very close. This indicates that having more

pieces in the control sequence can greatly help the Krotov algorithm to achieve higher fidelities despite limited strength of control fields.

The inset of Fig. 4B gives information on how the two points given by the vertical dashed line at N=20 connects when we expand the range of the control field. The bound is given as $1-J_{\rm max} \leq J_i \leq J_{\rm max}$. When $J_{\rm max}$ is increased from 0 to 20, the averaged fidelity from the Krotov method increases from 0.4 to above 0.8. This clearly demonstrates that the range of allowed values of control fields affects the outcome of the Krotov algorithm in a significant way.

We now proceed to consider the effect of discrete control to the averaged fidelities obtained by the algorithms. We start from SGD and Krotov with the range of the control field unrestricted, and the results are shown in Fig. 5. In both panels shown, we see that the averaged fidelities from the SGD method first increases for small M but quickly saturate. The insensitivity of the SGD against the discetization of the control field is due to the fact that SGD updates the control field moderately and can find sufficient control field values as desired within a relatively narrow range, even if the values are discretized. This is similar to the reason why the restriction on the range of control field has little effect on the results in Fig. 4(a).

On the other hand, the averaged fidelities from the Krotov method increase as functions of M, but the increase is much more pronounced for N=20 (Fig. 5B) than for N=6 (Fig. 5A). In Fig. 5B, the averaged fidelity from Krotov method exceeds that from SGD at around M+1=15. The result indicates that the successful implementation of the Krotov method depends crucially on the continuity of the problem, in terms of both the number of pieces in the control sequence as well as allowed values of the control field. We also note that at the limit $M\to\infty$, the extrapolated fidelity values are consistent to results shown in Fig. 2, providing a consistency check of our calculations.

After we have understood the performances of SGD and Krotov with the range of control fields unrestricted, we now compare all four algorithms considered in this work. QL and deep QL intrinsically favor a pre-set range of control field, and all calculations involving them are performed with $J_i \in [0,1]$. Therefore we also impose the same restriction to SGD and Krotov. The results are shown in Fig. 6. It is interesting to note that the averaged fidelities of QL and deep QL decreases with M, albeit not considerably. This is again because QL and deep QL favor a limited and concrete set of actions, and more choices will only add burden to the searching process, rendering the algorithms inefficient. For N=6 (Fig. 6A), QL and deep QL are comparable and are overall both better than SGD and Krotov, except for M+1>15in which the SGD becomes slightly better. For N=20(Fig. 6B), deep QL gives the best performance and QL the second, but for large M QL is not significantly better than the other two methods.

Before we conclude this section, we note that all results obtained have the target state being $|1\rangle$. Preparing a quantum state other than $|1\rangle$ may have different results, for which an example is presented in Sec. III of the Supplemental Material. Nevertheless, the overall observation of the pros and cons of the algorithms should remain similar.

Discussion

In this paper, we examined the performances of four algorithms, SGD, Krotov, QL and deep QL, on a simple problem of quantum state transfer. From the comparison, we can summarize the characteristics of the algorithms under different situations as follows.

- Dependence on the maximum number of pieces in the control sequence, N. When all algorithms are executed with the same number of iterations, the Krotov method performs the best when N is large enough. The fidelities of the other three methods decrease as N increases.
- Ability to adaptively segmenting. During the optimization process, QL and deep QL can adaptively reduce the number of pieces required and can thus find optimal solution efficiently. SGD and Krotov, on the other hand, have to work with a fixed number of N.
- Dependence on restricted strength ranges of the control field. QL and deep QL naturally work with restricted sets of actions so they perform well when the control field has restricted strength. Having the range of the strength bounded reduces the efficiency for both SGD and Krotov method, but the effect is moderate for SGD because its updates on the control field are essentially local. However, the Krotov method makes significant updates during its execution thus becomes severely compromised when the strength of the control field is limited.
- Ability to work with control fields taking M+1 discrete values. QL and deep QL again naturally work with discrete values of the control field. In fact, the fidelity from them decreases as the allowed values of the control field becomes more continuous (M increases). This problem may be circumvented using more sophisticated algorithms such as Actor-Critic [32], and the deep deterministic policy gradient method [33]. SGD is not sensitive to M because it works with a relatively small range of control field and a reasonable discretization is sufficient. The Krotov method, on the other hand, strongly favors continuously problem, i.e. M being large.
- Potential of generalization to high dimensions (more qubits). Although we only considered preparing a single qubit state in this work, we believe that except for QL, all other algorithms can be straightforwardly gen-

eralized to treat quantum control problems with more than one qubit. The Q-table maintained by QL quickly increases in dimensionality as more qubits are involved, making its application inefficient.

Moreover, we have found that the deep QL method, in general, has the best performance among the four algorithms considered, demonstrating the power of reinforcement learning in conjunction with neural networks in treating complex optimization problems.

Our direct comparison of different methods may also shed light on how these algorithms can be improved. For example, the Krotov method strongly favors the "continuous" problem, while Q-learning does not perform well. It should be possible that gradients in the Krotov method can be applied in the Q-learning procedure and thereby improves its performance. We hope that our work has elucidated the effectiveness of reinforcement learning in problems with different types of constraints, and in addition, it may provide hints on how these algorithm can be improved in future work.

Methods

In this section, we give a brief description of our implementation of QL and deep QL in this work. The full algorithms for all four methods used in this work are given in Sec. I of Supplemental Material.

Q-learning. For Q-learning, the key ingredients include a set of allowed states S, a set of actions A, and the reward r. A quantum state can be parametrized as

$$|\psi(\theta,\varphi)\rangle = \pm \left(\cos\frac{\theta}{2}|0\rangle + e^{i\varphi}\sin\frac{\theta}{2}|1\rangle\right),$$
 (3)

where (θ, ϕ) corresponds to a point on the Bloch sphere, and a possible global phase of -1 has been included. Our set of allowed states is defined as

$$S \equiv \{ |\psi(\theta, \varphi)\rangle | \theta \in s_{\theta}, \varphi \in s_{\varphi} \}, \tag{4}$$

where

$$s_{\theta} = \left\{ \frac{0\pi}{30}, \frac{1\pi}{30}, \dots, \frac{29\pi}{30} \right\}, \quad s_{\varphi} = \left\{ \frac{0\pi}{30}, \frac{1\pi}{30}, \dots, \frac{59\pi}{30} \right\}.$$
(5)

We note that this is a discrete set of states, and after each step in the evolution, if the resulting state is not identical to any of the member in the set, it will be assigned as the member that is closest to the state, i.e. having the maximum fidelity in their overlap.

In the *i*th step of the evolution, the system is at a state $s_i = |\psi_i\rangle \in \mathcal{S}$, and the action is given by the evolution operator $a_i = U_i = \exp\{-iH(J_i)dt\}$. All allowed values of the control field J_i therefore form a set of possible actions \mathcal{A} . The resulting state $U_i|\psi_i\rangle$ after this step is

then compared to the target state, and the reward is calculated using the fidelity between the two states as

$$r_i = \begin{cases} 10 & F \in (0.5, 0.9], \\ 100 & F \in (0.9, 0.999], \\ 5000 & F \in [0.999, 1], \end{cases}$$
(6)

so that the action that takes the state very close to the target is strongly rewarded. In practice, the agent chooses its action according to the ϵ -greedy algorithm [16], i.e., the agent either chooses an action with the largest Q(s,a) with $1-\epsilon$ probability, or with probability ϵ it randomly chooses an action in the set. The introduction of a nonzero but small ϵ ensures that the system is not trapped in a poor local minimum. The elements in Q-tables are then updated as:

$$Q(s_{i-1}, a_i) \leftarrow Q(s_{i-1}, a_i) + \alpha [r_i + \gamma \max_{a'} Q(s_i, a') - Q(s_{i-1}, a_i)], [7]$$
(7)

where a' refers to all possible a_i in this step, α is the learning rate, and γ is a reward discount to ensure the stability of the algorithm.

Deep Q-learning Deep Q-learning stores the actionvalue functions with a neural network Θ . Defining an agent state as

$$\mathbf{s} = \left[\operatorname{Re} \left(\langle 0 | \psi \rangle \right), \operatorname{Im} \left(\langle 0 | \psi \rangle \right), \operatorname{Re} \left(\langle 1 | \psi \rangle \right), \operatorname{Im} \left(\langle 1 | \psi \rangle \right) \right]^{T},$$
(8)

the network outputs the Q-value for each action $a \in \mathcal{A}$ as $Q(s, a; \Theta)$. We note that in deep QL, the discretization of states on the Bloch sphere is no longer necessary and we can deal with states that vary continuously. Otherwise the definitions of the set of actions and reward are the same as those in QL.

We adopt the double Q-network training approach [2], namely two neural networks, the evaluation network Θ and the target network Θ^- , are used for training. In the memory we store experiences defined as $e_i = (s_{i-1}, a_i, r_i, s_i)$. In each training step, an experience is randomly chosen from the memory, and the evaluation network is updated using the outcome derived from the experience.

We note that unlike the case for SGD and Krotov, in which the fidelity monotonically increases with more training in most cases, the fidelity output by QL and deep QL may experience considerable oscillations as the algorithm cannot guarantee optimal solutions in all trials. In this case, one just has to choose outputs which have higher fidelity as the learning outcome.

This work is supported by the Research Grants Council of the Hong Kong Special Administrative Region, China (Grant Nos. CityU 21300116, CityU 11303617, CityU 11304018), the National Natural Science Foundation of China (Grant Nos. 11874312, 11604277), and the Guangdong Innovative and Entrepreneurial Research

Team Program (Grant No. 2016ZT06D348).

- D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou,
 A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai,
 A. Bolton, et al., Nature 550, 354 (2017).
- [2] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al., Nature 518, 529 (2015).
- [3] M. Bukov, A. G. R. Day, D. Sels, P. Weinberg, A. Polkovnikov, and P. Mehta, Phys. Rev. X 8, 031086 (2018).
- [4] M. Bukov, arXiv preprint arXiv:1808.08910 (2018).
- [5] M. August and J. M. Hernández-Lobato, arXiv preprint arXiv:1802.04063 (2018).
- [6] F. Albarrán-Arriagada, J. C. Retamal, E. Solano, and L. Lamata, Phys. Rev. A 98, 042315 (2018).
- [7] X.-M. Zhang, Z.-W. Cui, X. Wang, and M.-H. Yung, Phy. Rev. A 97, 052333 (2018).
- [8] M. Y. Niu, S. Boixo, V. Smelyanskiy, and H. Neven, arXiv preprint arXiv:1803.01857 (2018).
- [9] T. Fösel, P. Tighineanu, T. Weiss, and F. Marquardt, Phys. Rev. X 8, 031084 (2018).
- [10] A. A. Melnikov, H. P. Nautrup, M. Krenn, V. Dunjko, M. Tiersch, A. Zeilinger, and H. J. Briegel, Proc. Natl. Acad. Sci., 201714936 (2018).
- [11] V. Dunjko, J. M. Taylor, and H. J. Briegel, Phys. Rev. Lett. 117, 130501 (2016).
- [12] A. Hentschel and B. C. Sanders, Phys. Rev. Lett. 104, 063603 (2010).
- [13] A. G. Day, M. Bukov, P. Weinberg, P. Mehta, and D. Sels, Phys. Rev. Lett. 122, 020601 (2019).
- [14] R.-B. Wu, B. Chu, D. H. Owens, and H. Rabitz, Phys. Rev. A 97, 042122 (2018).
- [15] C. Ferrie, Phys. Rev. Lett. 113, 190404 (2014).
- [16] R. S. Sutton and A. G. Barto, Reinforcement learning: An introduction (MIT press Cambridge, 1998).
- [17] L. Wang, Phys. Rev. B 94, 195105 (2016).
- [18] G. Carleo and M. Troyer, Science **355**, 602 (2017).
- [19] D.-L. Deng, X. Li, and S. Das Sarma, Phys. Rev. X 7, 021021 (2017).
- [20] J. Carrasquilla and R. G. Melko, Nat. Phys. 13, 431 (2017).
- [21] J. Li, X. Yang, X. Peng, and C.-P. Sun, Phys. Rev. Lett. 118, 150503 (2017).
- [22] Y.-T. Hsu, X. Li, D.-L. Deng, and S. Das Sarma, Phys. Rev. Lett. 121, 245701 (2018).
- [23] V. F. Krotov, Global methods in optimal control theory (Marcel Dekker Inc., New York, 1996).
- [24] J. R. Petta, A. C. Johnson, J. M. Taylor, E. A. Laird, A. Yacoby, M. D. Lukin, C. M. Marcus, M. P. Hanson, and A. C. Gossard, Science 309, 2180 (2005).
- [25] A. Greilich, S. E. Economou, S. Spatzek, D. Yakovlev, D. Reuter, A. Wieck, T. Reinecke, and M. Bayer, Nat. Phys. 5, 262 (2009).
- [26] E. Poem, O. Kenneth, Y. Kodriano, Y. Benny, S. Khatsevich, J. E. Avron, and D. Gershoni, Phys. Rev. Lett. 107, 087401 (2011).
- [27] A. Y. Kitaev, A. Shen, and M. N. Vyalyi, Classical and quantum computation (American Mathematical Society, 2002).

- [28] A. W. Harrow, B. Recht, and I. L. Chuang, J. Math. Phys 43, 4445 (2002).
- [29] E. T. Campbell, B. M. Terhal, and C. Vuillot, Nature 549, 172 (2017).
- [30] T. Caneva, M. Murphy, T. Calarco, R. Fazio, S. Montangero, V. Giovannetti, and G. E. Santoro, Phys. Rev. Lett. 103, 240501 (2009).
- [31] M. Murphy, S. Montangero, V. Giovannetti, and T. Calarco, Phys. Rev. A 82, 022318 (2010).
- [32] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lilli-
- crap, T. Harley, D. Silver, and K. Kavukcuoglu, in *International Conference on Machine Learning* (2016) pp. 1928–1937.
- [33] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, arXiv preprint arXiv:1509.02971 (2015).

Supplementary material

I. PSEUDO CODE FOR THE ALGORITHMS

Here, we provide the pseudo code for all four algorithms used in the main text.

Algorithm 1: Stochastic Gradient Descendent (SGD).

```
Set initial guess of \boldsymbol{J} = [J_1, J_2, \cdots, J_N]^T randomly for iteration = 1, 500

Generate a random unit vector \boldsymbol{v}

Set \boldsymbol{J}^+ = \boldsymbol{J} + \alpha \boldsymbol{v}, \boldsymbol{J}^- = \boldsymbol{J} - \alpha \boldsymbol{v}

Calculate the gradient \boldsymbol{g} = \frac{F(\boldsymbol{J}^+) - F(\boldsymbol{J}^-)}{2\alpha}

Update \boldsymbol{J} \leftarrow \boldsymbol{J} - \beta \boldsymbol{g}

restrict \boldsymbol{J} to the range [J_{\min}, J_{\max}]

end for
```

Algorithm 2: Krotov algorithm.

Initialize J arbitrarily

Calculate and store $|\psi_i\rangle$ at each step i according to $|\psi_i\rangle=e^{-iH_it}|\psi_{i-1}\rangle$

Set co-state at step N as $|\chi_N\rangle = |\phi\rangle \langle \phi|\psi_N\rangle$

Calculate and store $|\chi_i\rangle$ at each step i according to $|\chi_{i-1}\rangle=e^{iH_it}|\psi_i\rangle$

```
 \begin{split} \textbf{for iteration} &= 1,500 \\ \textbf{for } i &= 1, \ N \\ &\quad \text{Calculate } |\psi_i\rangle \text{ according to } |\psi_i\rangle = e^{-iH_it}|\psi_{i-1}\rangle \\ &\quad \text{Update } J_i \leftarrow J_i + \text{Im}\langle\chi_i|\partial_J H(J_i)|\psi_i\rangle \\ \textbf{end for} \\ &\quad \text{Fidelity } F \leftarrow |\left\langle\phi|\psi_N\right\rangle|^2 \\ \end{split}
```

Set co-state at step N as $|\chi_N\rangle = |\phi\rangle \langle \phi|\psi_N\rangle$

Calculate and store $|\chi_i\rangle$ at each step i according to $|\chi_{i-1}\rangle=e^{iH_it}|\psi_i\rangle$

restrict \boldsymbol{J} to the range $[J_{\min}, J_{\max}]$

end for

Algorithm 3: Q-learning.

```
Initialize Q(s, a) = 0 for all s \in \mathcal{S}, a \in \mathcal{A}.
Initialize s_0 = |\psi_0\rangle = |0\rangle
for iteration = 1,500
for i = 1, N
```

```
With \epsilon probability choose a_i randomly, otherwise a_i = \underset{a}{\arg\max} Q(s_{i-1}, a)

Take the action a_i, evaluate the reward r_i and |\psi_i\rangle

Set s_i as the nearest point in \mathcal S to |\psi_i\rangle

Update Q(s_{i-1}, a_i) according to Eq. (7)

Break if 1 - F < 10^{-3}

end for
```

Algorithm 4: Deep Q-learning.

Initialize memory R as empty

end for

end for

Initialize the evaluation network $\Theta,$ and target network $\Theta^- \leftarrow \Theta$

```
for iteration = 1, 500

Initialize |\psi\rangle = |0\rangle and \mathbf{s_0} = [1, 0, 0, 0]^T

for i = 1, N, \mathbf{do}
```

With ϵ probability choose a_i randomly, otherwise $a_i = \arg\max Q(s_{i-1}, a; \Theta)$

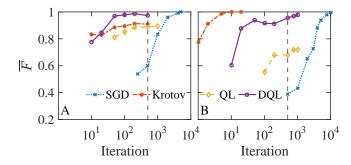
Take the action a_i , and evaluate the reward r_i and state $\boldsymbol{s_i}$

Store experience $e_i = (\mathbf{s}_{i-1}, a_i, r_i, \mathbf{s}_i)$ in memory Rif t is divisible by t_{learn} Sample minibatch of experiences e_k Set $y_k = r_k + \gamma \max_{a'} \hat{Q}(\mathbf{s}_k, a'; \Theta^-)$ Update Θ by minimizing $L = [y_k - Q(\mathbf{s}_{k-1}, a_k; \Theta)]^2$ end if

Every C times of learning, set $\Theta^- \leftarrow \Theta$ Break if $1 - F < 10^{-3}$

II. IMPROVING THE FIDELITY WITH MORE ITERATIONS

In Fig. 2 of the main text we have compared four algorithms in terms of the average fidelity versus the maximum number of control pieces. To ensure a fair comparison, all algorithms are requested to stop at $N^{\text{iter}} = 500$,

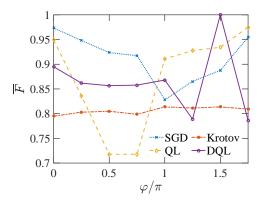


Supplementary Figure S1: Average fidelity versus number of iterations N^{iter} . (A) maximum number of control pieces N=20 and (B) N=50. \overline{F} is obtained by averaging over 100 runs. The vertical grey dashed lines at $N^{\mathrm{iter}}=500$ in both panels correspond to the results shown in Fig. 2 of the main text with the respective N values.

i.e. after 500 iterations. Here we show that by allowing more iterations, the fidelity of all algorithms can improve, and the improvement is particularly pronounced for the SGD method. Fig. S1 shows the average fidelity versus the number of iterations, with all other parameters and constraints the same as those used in Fig. 2 of the main text. Fig. S1A shows the results at N=20 for which SGD has a relatively low fidelity (around 0.6 at $N^{\text{iter}} = 500$). As the iteration continues, the fidelity of SGD improves substantially, reaching 1 at $N^{\text{iter}} \gtrsim 500$. On the other hand, the fidelities for other methods do not change much as the number of iteration is improved. Fig. S1A shows the results at N=50. We see that results from the Krotov method reaches 1 at $N^{\rm iter} \sim 20$, those from DQL improves slightly after $N^{\text{iter}} = 50$, but again the increase of fidelity is most pronounced for the SGD method, with the fidelity increasing from 0.4 at $N^{\text{iter}} = 500 \text{ to } 1 \text{ at } N^{\text{iter}} = 10000.$ It is also interesting to note that the fidelity output from QL does not carry a considerable increase, likely because the non-zero failure rate cannot be decreased simply by adding more iterations. We therefore conclude that (1) The result from SGD is most sensitive to the total number of iterations: the fidelity can reach 1 as long as one iterate the algorithm long enough. However, this process could be very resource-consuming compared to other methods that can have high fidelity values for a much smaller number of iterations. (2) The QL method is most insensitive to more iterations as its intrinsic failure rate cannot be suppressed this way. Adding more iterations will not increase its fidelity output by a notable amount.

III. TARGET STATE DEPENDENCE OF THE LEARNING OUTCOME

In all results shown in the main text, our target state is always $|1\rangle$. In order to provide a more complete picture,



Supplementary Figure S2: The dependence of the learning outcome (average fidelity) on φ , which parametrizes different target states. The target state is defined in Eq. (S-1) with φ as its sole parameter.

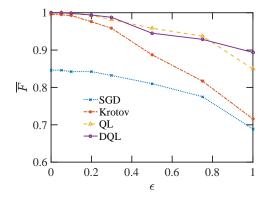
we take states on the equator of the Bloch sphere

$$|\phi\rangle = \frac{1}{\sqrt{2}} (|0\rangle + e^{i\varphi}|1\rangle),$$
 (S-1)

as examples of other possible target states. This set of states has one sole parameter φ so that one may plot the average fidelity versus φ in a figure, which we show in Fig. S2 (Note that $N^{\text{iter}} = 500$). First, we see that the Krotov method is not sensitive to the change of states, and the fidelity is maintained at about 0.8 for all target states concerned. All other three methods have outputs that vary a lot with the change of target states. Taking deep QL as an example, the average fidelity reaches 1 for $\varphi = 1.5\pi$, but is lower than 75% for $\varphi = 1.25\pi$. While it may not be meaningful to provide a full explanation, we attribute the variance as the result of a discretized action space, i.e. the allowed actions are limited and cannot cover all points on the Bloch sphere. We note that developments are on-going in order to allow reinforcement learning to choose action from a continuous set [32, 33]. Their implications on quantum physics warrant further studies.

IV. NOISE EFFECT

Here, we briefly discuss the robustness of the control sequences found by different algorithms against noises. We first generate the control fields without noises, choosing one for each algorithm that possesses the highest fidelity from 20 runs. Then we feed noise into the evolution via the control field: $\tilde{J}_i \to J_i + \delta J_i$, where the error term δJ_i is uniformly drawn from $[-\epsilon, \epsilon]$ with ϵ the noise level. Then for each control sequence we produce a set of 100 realizations of noises, and the resulting fidelities are averaged. In Fig. S3 we show the results. As is clear from the figure, the average fidelities drop most significantly



Supplementary Figure S3: Average fidelity versus the noise level in different algorithms. The target state is $|1\rangle$, N=20. Each point of \overline{F} is an average over 100 runs.

for Krotov and SGD, but only moderately for QL and DQL. We believe that it is because the control sequences produced by QL and DQL are less complex, i.e. having shorter time durations and smaller jumps in the values of the control fields.