Data-driven gradient algorithm for high-precision quantum control

Re-Bing Wu[*]
Department of Automation, Tsinghua University, Beijing 100084, China
and Center for Quantum Information Science and Technology, BNRist, Beijing 100084, China

Bing Chu[†]
School of Electronic and Computer Science, University of Southampton, Southampton SO17 1BJ, United Kingdom

David H. Owens[‡]
Department of Automation, Zhengzhou University, Zhengzhou 450001, China
and Department of Automatic Control and Systems Engineering, The University of Sheffield, Mappin Street, Sheffield S1 3JD, United Kingdom

Herschel Rabitz[§]
Department of Chemistry, Princeton University, Princeton, New Jersey 08544, USA

(Received 9 January 2018; published 24 April 2018)

In the quest to achieve scalable quantum information processing technologies, gradient-based optimal control algorithms (e.g., GRAPE) are broadly used for implementing high-precision quantum gates, but their performance is often hindered by deterministic or random errors in the system model and the control electronics. In this paper, we show that GRAPE can be taught to be more effective by jointly learning from the design model and the experimental data obtained from process tomography. The resulting data-driven gradient optimization algorithm (d-GRAPE) can in principle correct all deterministic gate errors, with a mild efficiency loss. The d-GRAPE algorithm may become more powerful with broadband controls that involve a large number of control parameters, while other algorithms usually slow down due to the increased size of the search space. These advantages are demonstrated by simulating the implementation of a two-qubit controlled-NOT gate.

DOI: 10.1103/PhysRevA.97.042122

I. INTRODUCTION

In practical quantum information processing, high-precision implementation of universal quantum gates (usually involving 1–3 qubits) is vital. Although the current control technology has been able to meet the minimum requirement for quantum error correction [1] (e.g., the 0.6%–1% error threshold for surface codes has been reached in superconducting circuits [2], ion traps [3], quantum dots [4], and nitrogen-vacancy centers in diamond [5]), the achievable precision still needs to be improved in order to reduce the resource overhead required for scalable quantum computation [6].

Towards this "last mile" target, an effective method for gate tuneup is to optimize the control pulses by following the gradient direction of the error function, which popularly has one form known as the GRAPE (gradient ascending pulse engineering) algorithm [7]. When supplied with abundant control resources, the algorithm is highly efficient in that the optimization almost always quickly converges to a global optimal solution, owing to the underlying expectation of finding an attractive trap-free optimal control landscape [8–10]. The GRAPE algorithm is by nature *offline* (or *ex situ* [11]) because the optimization is usually with respect to a design model identified from prior experiments, and no real online data is used during the optimization process. Thus, the systematic errors in the design model (e.g., the identified Hamiltonian and the pulse distortion by a waveform generater), as well as the uncharacterized random noises in the system and pulses, limit the control precision. Regarding these items, the designed control pulses should be immune to the systematic errors and be robust to the random noises.

Online (or *in situ*) learning can in principle correct for the systematic errors by iteratively calibrating the control pulses based on measurement outcomes. This learning control concept can be traced back to the early 1990s in the control of molecules by training ultrafast laser pulses [12], which has been successful in hundreds of physical and chemical experiments [13]. In most applications, the control objective is with respect to a target state or the ensemble average of some quantum observable where the control fields are updated by heuristic optimization algorithms such as a genetic algorithm [12] or evolutionary strategy [14]. Learning control for quantum gate tune-up is much more difficult than the aforementioned applications, because the full characterization of the control outcome requires a process tomography that needs many more experiments to measure additional observables at high precision. In existing protocols, the extra data acquisition problem is usually bypassed via randomized

[*]rbwu@tsinghua.edu.cn
[†]b.chu@soton.ac.uk
[‡]d.h.owens@shef.ac.uk
[§]hrabitz@Princeton.Edu

benchmarking (RB) [15], which is much easier for gate error verification without having to fully reconstruct the gate matrix. Several RB-based learning algorithms have been proposed, e.g., the Nelder-Mead algorithm was used in [16] and [17], with applications to superconducting qubits. To exploit the attractive trap-free control landscape [8], gradient-based (or greedy) algorithms were also introduced to accelerate the online optimization, where extra measurements (proportional to the number of control variables) need to be done to estimate the full or partial gradient from the data [11,18–21].

The complexity of online learning control algorithms mainly depends on the total experimental costs, while the numerical calculations on a computer are usually negligible when only a few qubits are involved. In the existing algorithms, the overall cost can be very high due to the required many iterations (mainly for RB-based optimization [11,16,17]) or the expensive measurements in each iteration (mainly for gradient-based optimization [18–20]).

To further reduce the total experimental cost, we find that the design model, which is often used for obtaining a good initial guess for the control pulse, can play a new role in accelerating the succeeding online learning calibration process. This opportunity arises because the design model contains valuable *a priori* knowledge about the experimental system, which is obtained from elaborately designed offline measurements. This motivation leads to the algorithm proposed in this paper, in which the design model is embedded into the data-driven learning procedure to synthesize the gradient vector also utilizing data from process tomography. The algorithm can effectively reduce the number of iterations by predicting the gradient descent or ascent direction for quantum gate tuneup, which compensates for the increased cost of tomography. Besides, under circumstances where broad-bandwidth controls are required for noise suppression or high-speed gate operations, the total experimental cost of our algorithm may be further reduced, while the corresponding cost usually increases with other algorithms. We refer to the method presented here as a data-driven type of GRAPE algorithm, or d-GRAPE for short. The remainder of the paper is organized as follows. The d-GRAPE algorithm is described in Sec. II, whose effectiveness in correcting model error and control pulse distortion is demonstrated through simulations in Sec. III. Finally, conclusions are presented in Sec. IV.

II. THE DATA-DRIVEN GRADIENT ALGORITHM

In this section, we will present the basic procedure of the data-driven gradient algorithm.

A. The quantum control model

We assume that the quantum control system is closed and governed by the following Schrödinger equation:

$$\dot{U}(t) = -i\left[H_0 + \sum_{k=1}^{m} u_k(t)H_k\right]U(t), \tag{1}$$

where $U(t) \in \mathbb{C}^{N\times N}$ represents the quantum gate operation on the states, with $U(0) = \mathbb{I}_N$ the identity matrix, and $u_k(t) \in \mathbb{R}$, $k = 1\ldots,m$, are the control fields imposed on the control system. The free Hamiltonian H_0 and the control Hamiltonians H_k's are $N \times N$ Hermitian matrices that steer the unitary $U(t)$.

In practice, the above Hamiltonians are never precisely known. Thus, any numerical calculation has to be based on a design model,

$$\dot{U}_D(t) = -i\left[H_{D,0} + \sum_{k=1}^{m} v_k(t)H_{D,k}\right]U_D(t), \tag{2}$$

that can be accessed by a computer. The free and control Hamiltonians in the design model (2) can be very close to those in the actual system (1), but they are always imprecise to some degree. The control pulses $v_k(t)$ in the design model are often chosen as piecewise-constant pulses to facilitate numerical simulation on a digital computer and in some experimental situations. Note that the (designed) control pulses $v_k(t)$ are usually not identical with the actual pulses $u_k(t)$ applied to the system, because the control signal produced by an arbitrary waveform generator (AWG) is often distorted due to various factors, including electronic limitations and transmission through the control line to the qubit. Such distorted signals have rising and falling edges or other unanticipated features, which are sometimes called quantum gate bleedthrough [17]. For example, the distortion can be modeled by a linear filter described as follows:

$$u_k(t) = \mathcal{D}[v_k(t)] = \int_0^\infty h(t-\tau)v_k(\tau)d\tau, \tag{3}$$

where $h(t)$ is the impulse response of the linear filter. The control pulse is distortion free only when $h(t) = \delta(t)$ is the Dirac function. In the following, we will show how to correct these errors by learning from online data.

B. From GRAPE to d-GRAPE

The goal of quantum gate tune-up is to find proper design control pulse sequences $\{v_k(t)\}$ such that the generated control $\{u_k(t)\}$ can lead the system propagator $U(T)$ as close as possible to a desired unitary matrix U_f. This can be achieved by minimizing the *infidelity* function [22]

$$\mathcal{J} = \frac{1}{2N}\|U(T) - U_f\|^2, \tag{4}$$

where the norm is defined as $\|X\| = \sqrt{\mathrm{Tr}(X^\dagger X)}$.

There are different ways of utilizing the gradient to optimize the control pulse. We illustrate the concept in the paper with the typical steepest descent algorithm that updates the control pulses in the following fashion:

$$v_k(t,\ell+1) = v_k(t,\ell) - \alpha(\ell)\cdot g_k(t,\ell), \tag{5}$$

where $g_k(t,\ell) = \frac{\delta\mathcal{J}}{\delta v_k(t,\ell)}$ is the gradient in the ℓth iteration and $\alpha(\ell)$ is the learning rate that is chosen as a sufficiently small positive real number. Taking $U(T)$ as an implicit function of $\{v_k(t)\}$ through (1) and (3), we have

$$\begin{aligned} g_k(t,\ell) &= \int_0^\infty \frac{\delta\mathcal{J}}{\delta u_k(t',\ell)}\frac{\delta u_k(t',\ell)}{\delta v_k(t,\ell)}dt' \\ &= \int_0^\infty \langle\Delta(T,\ell), H_k(t',\ell)\rangle\frac{\delta u_k(t',\ell)}{\delta v_k(t,\ell)}dt', \end{aligned}$$

where $H_k(t,\ell) = U^\dagger(t,\ell)H_k U(t,\ell)$ and the inner product is defined as $\langle X,Y\rangle = \mathrm{Tr}(X^\dagger Y)$. The error matrix is

$$\Delta(T,\ell) = \frac{1}{2i}[U_f^\dagger U(T,\ell) - U^\dagger(T,\ell)U_f].$$

The variation term in the integral is induced by the distortion of the control pulses. In the linear case exemplified in (3), we have

$$\frac{\delta u_k(t',\ell)}{\delta v_k(t,\ell)} = h(t'-t). \tag{6}$$

Because the true gradient function (6) can never be precisely evaluated due to the unavailability of the true model of the system, a practical operation is to ignore the pulse distortion and calculate the gradient in an *offline fashion*, as follows:

$$g_k^{\mathrm{OL}}(t,\ell) = \langle \Delta_D(T,\ell), H_{D,k}(t,\ell)\rangle, \tag{7}$$

where $H_{D,k}(t,\ell) = U_D^\dagger(t,\ell)H_{D,k}U_D(t,\ell)$ and

$$\Delta_D(T,\ell) = \frac{1}{2i}[U_f^\dagger U_D(T,\ell) - U_D^\dagger(T,\ell)U_f]$$

are both computed from the design model. Since the optimization is completely blind without checking the control performance with experimental data, the learning process along this gradient direction will be inevitably guided to a false solution that is optimal for the design model but not for the actual system.

To find the genuine optimal control pulses, we take advantage of both of the above two approaches. The key concept is to estimate the gradient as follows:

$$\hat{g}_k(t,\ell) = \langle \hat{\Delta}(T,\ell), H_{D,k}(t,\ell)\rangle, \tag{8}$$

where the error matrix $\hat{\Delta}(T,\ell)$ comes form the estimation of $\Delta(T,\ell)$ through process tomography of $U(T)$, and $H_{D,k}(t,\ell)$ is calculated from the design model as in (7). In this way, the real data are employed in order to deduce whether the learning algorithm is converging to a correct solution such that $\Delta(T,\ell) = 0$, and its incorporation with $H_{D,k}(t,\ell)$ provides an approximate gradient whose deviation from the real gradient depends on the accuracy of the design model. The entire learning process is shown in Fig. 1, where the explicit use of the design model is the major difference with existing *model-free* learning control strategies in the literature. Therefore, we refer to the algorithm as d-GRAPE.

C. Convergence analysis

It is difficult to rigorously prove the convergence of the d-GRAPE to a globally optimal solution. Heuristically, d-GRAPE should converge to at least a locally optimal solution because the estimated gradient (8) can still maintain descent, although possibly not the steepest in the presence of various uncertainties, as long as they are not too large. On the other hand, d-GRAPE can stop at a desired globally optimal control solution corresponding to $U(T) = U_f$ (assuming that the tomography error is negligible), where the gradient (8) vanishes. Therefore, when the system is controllable and the control resources are sufficiently abundant [9], the well-preserved attractive character of the control landscape should assure that d-GRAPE almost always converges to the desired global optimal solution,

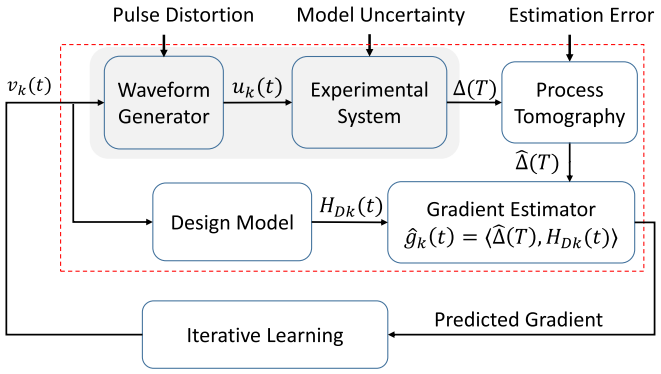

FIG. 1. Schematic diagram of the data-driven GRAPE (d-GRAPE) optimization procedure. The gradient is estimated from both the design model and the online data (for process tomography), which can in principle correct all deterministic errors such as the pulse distortion and the model uncertainty.

which will be verified in the following simulation examples. In principle, d-GRAPE is able to correct for any deterministic errors in the model or in the control pulses. Its precision is limited by that of the process tomography and other random noise sources in the system.

Compared with the existing online learning algorithms, d-GRAPE will be more competitive when broadband controls that involve a large number of variables are required for high precision, speed, and robustness [23,24]. Under such circumstances, the experimental cost of d-GRAPE per iteration will stay invariant, but the convergence may be faster owing to increased freedom in the control. However, the RB-based algorithms are expected to be more expensive because many more iterations are needed for searches in the enlarged control space, as well as for the gradient-based algorithms proposed in [18–20], whose experimental costs per iteration increase with the number of control parameters.

III. SIMULATIONS

In this section, we will show by numerical simulations how the algorithm can correct deterministic errors in the model and control pulses.

A. Simulation model

We assume that the actual system is described by the following Hamiltonian:

$$H(t) = J\sigma_z^1 \otimes \sigma_z^2 + \sum_{i=1}^{2}\left[u_x^i(t)\sigma_x^i + u_y^i(t)\sigma_y^i\right],$$

where J is the coupling strength between the two qubits. The design model is as follows:

$$H_D(t) = (J+\delta J)\sigma_z^1 \otimes \sigma_z^2 + \sum_{i=1}^{2}\left[v_x^i(t)\sigma_x^i + v_y^i(t)\sigma_y^i\right],$$

in which δJ represents the identification error of J in the design model.

Moreover, we assume that the control pulses are distorted by a linear filter

$$u^i_{x,y}(t) = \int_0^t h(t-\tau)v^i_{x,y}(\tau)d\tau, \quad i = 1,2,$$

in which the impulse response $h(t)$ is taken as

$$h(t) = \frac{1}{t_r}e^{-t/t_r}, \quad t \geqslant 0. \tag{9}$$

The time constant t_r characterizes the degree of pulse distortion by the steepness of the rising edge of distorted pulses. The pulses are heavily distorted when t_r is long.

B. Gate tuneup simulation results

To demonstrate the ability of quantum gate tune-up by d-GRAPE, we test the target of a controlled-NOT gate

$$U_f = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 \end{pmatrix}.$$

In the simulation, we set the coupling constant as $J = 1$ and the final time as $T = 5$. The time interval is evenly divided into $M = 20$ subintervals, and hence the duration of each subinterval is $\Delta t = T/M$.

In Fig. 2, we show three cases with parametric error δJ in J and pulse distortion characterized by st_r. Each case includes results from 12 different initial random guesses. We first offline optimize these fields [i.e., following $g_k^{\mathrm{OL}}(t)$ in (7)] to obtain a set of candidate pulses that are close to the optimal solution. Then, starting from these pulses, we perform d-GRAPE [i.e., following $\hat{g}_k(t)$ in (8)] based on the Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm (a very popular gradient-based optimization algorithm [25]), which is more efficient than the steepest descent gradient algorithm. The estimation errors in the process tomography are simulated by injecting an additive random noise $\Delta U(\ell)$ (whose Frobenius norm is $\sim 2 \times 10^{-5}$) to $U(T,\ell)$ in each iteration. For comparison, we also run the ideal GRAPE [i.e., following $g_k(t)$ in (6), assuming that both δJ and t_r are precisely known] from the same set of initial pulses.

The simulation results show that the precision of the candidate pulses obtained from offline optimization (at the beginning of the optimization process shown in the plots) is always limited by the accuracy of the design model. When the model error is relatively small [see Fig. 2(a)], the succeeding optimization based on the proposed d-GRAPE algorithm (solid curves) almost always converges to its global optimal solution that is limited by the tomography error. Compared with the ideal GRAPE optimization (see the blue dash curves), its convergence speed is only slightly reduced.

When the model error is not small enough [e.g., with severe pulse distortion in Fig. 2(b) or parameter deviation δJ in Fig. 2(c)], fewer runs can quickly converge to the global optimal solution. Some runs still converge, but at the price of an increase in the number of iterations. We plot in Fig. 3 the shapes of the corrected and actual x-axis control signals on the first qubit, showing that the d-GRAPE can correct for large pulse

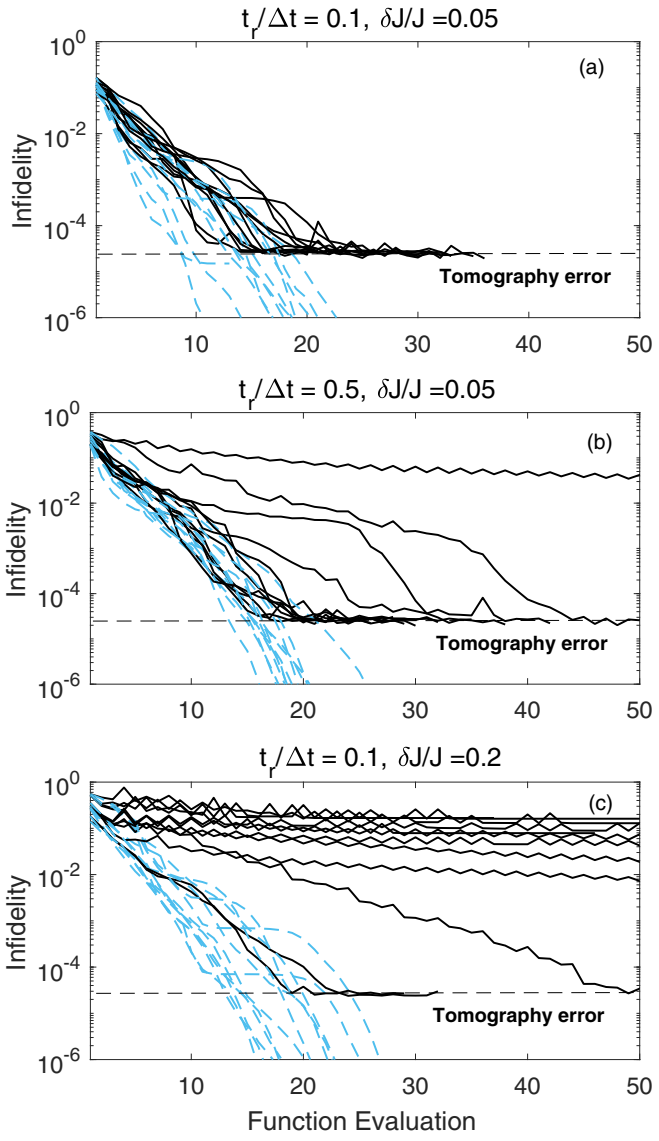

FIG. 2. Two-qubit quantum gate tuneup with the proposed d-GRAPE algorithm (solid black curves) for different model uncertainties and their comparison with an ideal GRAPE algorithm (blue dashed curves). Each case include 12 runs from different initial guesses. (a) Small pulse distortion $t_r/\Delta t = 0.1$ and small parametric error $\delta J/J = 0.05$; (b) large pulse distortion $t_r/\Delta t = 0.5$ and small parametric error $\delta J/J = 0.05$; and (c) small pulse distortion $t_r/\Delta t = 0.1$ and large parametric error $\delta J/J = 0.20$. The ultimate control precision $\sim 2 \times 10^{-5}$ is limited by the estimation error of the process tomography (indicated by the horizontal dashed line).

distortion to achieve high-precision control without having to exactly know how the pulses are distorted.

The model errors we choose in the simulations are relatively large (e.g., 20% error in J and pulse distortion $t_r/\Delta t = 0.5$), and even under such a bad situation d-GRAPE can still tune up the gate to some extent. When the model error gets larger, more and more optimizations become slower, and there even exist cases that the d-GRAPE algorithm gets lost and is trapped at a local false optimum solution. Thus, d-GRAPE should not be applied with a very coarse model because of the potential traps

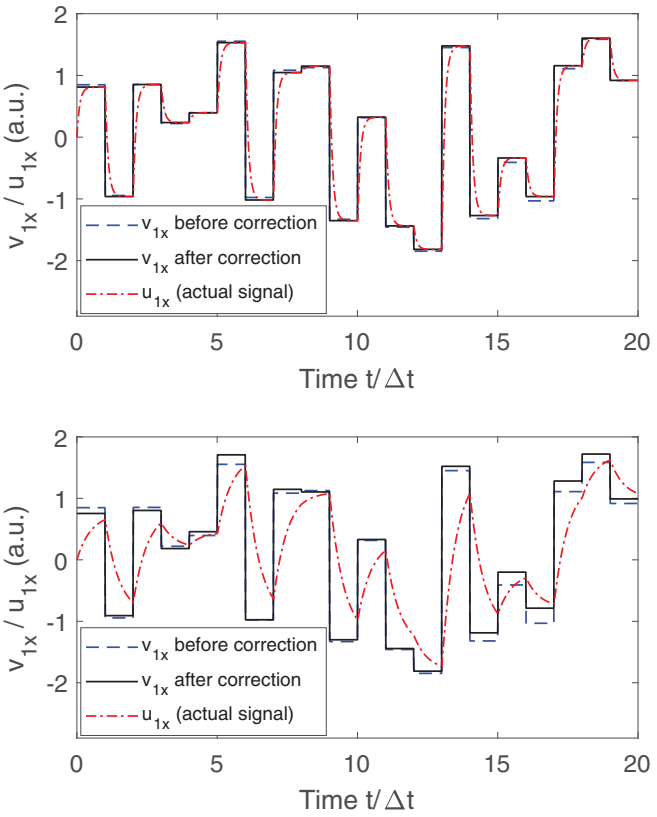

FIG. 3. The examples of optimized control pulses on the first qubit along the x axis with $\delta J/J = 0.1$. The pulse distortion parameters are $t_r/\Delta t = 0.1$ (upper plot) and $t_r/\Delta t = 0.5$ (lower plot). The blue dashed curves are the initial AWG reference signal, and the black solid curves are the corrected AWG signal optimized to the precision $\sim 2 \times 10^{-5}$ that is limited by the tomography error. The actual distorted signals with rising and falling edges are shown by red dash-dotted curves.

and the increased experimental cost on process tomography. In practice, one should improve the precision of the design model as much as possible. Based on the high-precision model, the d-GRAPE algorithm can correct the error caused by the residue model imprecision within a few iterations.

C. Comparison with RB-based algorithms

We also tested the performance of d-GRAPE using different numbers of control pulses and compared its performance with that of the gradient-free Nelder-Mead (NM) algorithm. The latter algorithm can be applied based on randomized benchmarking without having to use process tomography. The simulations are all based on a relatively precise model with $\delta J/J = 0.02$.

As shown in Fig. 4, when there are few pulses to tune ($M = 10$), d-GRAPE is trapped over a very rugged control landscape (i.e., resulting from an insufficient number of control variables producing false landscape traps), while NM can struggle to achieve a high precision after about 2000 iterations. Given more control pulses, d-GRAPE can easily find high-precision control solutions over an almost trap-free control landscape in several tens of iterations. Correspondingly, the

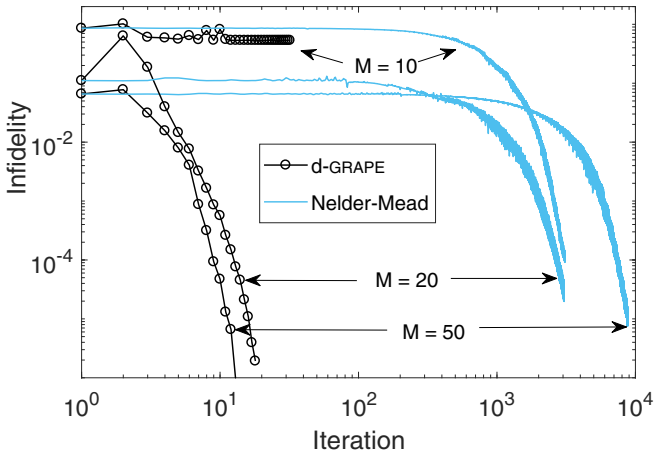

FIG. 4. Performance comparison of the d-GRAPE algorithm and the Nelder-Mead algorithm. d-GRAPE may fail when there are only a few control parameters ($M = 10$) and succeed with more control parameters ($M = 20$ and $M = 50$). When M increases from 20 to 50, d-GRAPE becomes more efficient because fewer iterations are required, while the Nelder-Mead algorithm takes much more iterations to converge.

number of NM iterations is hundreds of times (or even over 1000 times) larger than that of d-GRAPE iterations. More importantly, the number of iterations increases for NM but decreases for d-GRAPE when the number of control pulses grows. Hence, d-GRAPE is supposed to outperform NM when the number of control variables is sufficiently large, which is expected to be the case when higher precision and robustness are demanded.

IV. CONCLUSION AND DISCUSSION

To summarize, we have proposed a data-driven gradient (d-GRAPE) algorithm for optimizing laboratory control pulses against deterministic errors. The entire optimization procedure essentially performs both in a reinforcement learning manner from the online data in addition to supervised learning from the design model (or offline data). Analyses and simulations exemplify the calibration ability against errors induced by pulse distortion and model uncertainty, which is in principle extendable to more general nonuniform and nonlinear errors, as long as the process tomography can be done with sufficient precision and a reasonably good design model is available.

There is much room for the d-GRAPE algorithm to be improved. Several extensions of the algorithm are possible. First, extracting more knowledge from the offline model will improve the online optimization. For example, we can estimate the gradient more precisely by incorporating the pulse distortion function $h(t)$ that can be offline identified from the waveform generator, or we can use a more sophisticated learning algorithm such as a Newton algorithm, because the Hessian matrix can be estimated based on the same use of process tomography without increasing the number of experiments. Second, combined with adaptive tomography [26–28], it is possible to simultaneously improve the precision of the control and the process tomography, which will further accelerate the learning process.

We also remark that the d-GRAPE algorithm can be extended to more general objectives, e.g., quantum state preparation problems, where the cost of state tomography is cheaper and hence can be more efficient. When the real quantum system undergoes open dynamics, we can replace the unitary propagators by open-system process matrices, but the achievable precision may be limited by the decoherence effects. These potential topics and developments will be explored in the future.

ACKNOWLEDGMENTS

The author R.B.W. acknowledges support from the NSFC (Grants No. 61773232, No. 61374091, and No. 61134008) and the National Key Research and Development Program of China (Grant No. 2017YFA0304300). The author H.R. acknowledges support from the ARO (W911NF-16-1-0014). The authors also acknowledge useful discussions with Professor Xinhua Peng and Dr. Xiaodong Yang from the University of Science and Technology of China.

[1] C. H. Bennett, D. P. DiVincenzo, J. A. Smolin, and W. K. Wootters, Phys. Rev. A **54**, 3824 (1996).
[2] R. Barends, J. Kelly, A. Megrant, A. Veitia, D. Sank, E. Jeffrey, T. C. White, J. Mutus, A. G. Fowler, B. Campbell *et al.*, Nature (London) **508**, 500 (2014).
[3] T. P. Harty, D. T. C. Allcock, C. J. Ballance, L. Guidoni, H. A. Janacek, N. M. Linke, D. N. Stacey, and D. M. Lucas, Phys. Rev. Lett. **113**, 220501 (2014).
[4] M. Veldhorst, J. C. C. Hwang, C. H. Yang, A. W. Leenstra, B. de Ronde, J. P. Dehollain, J. T. Muhonen, F. E. Hudson, K. M. Itoh, A. Morello *et al.*, Nat. Nanotechnol. **9**, 981 (2014).
[5] X. Rong, J. Geng, F. Shi, Y. Liu, K. Xu, W. Ma, F. Kong, Z. Jiang, Y. Wu, and J. Du, Nat. Commun. **6**, 8748 (2015).
[6] S. J. Devitt, W. J. Munro, and K. Nemoto, Rep. Prog. Phys. **76**, 076001 (2013).
[7] N. Khaneja, T. Reiss, C. Kehlet, T. Schulte-Herbruggen, and S. J. Glaser, J. Magn. Reson. **172**, 296 (2005).
[8] H. A. Rabitz, M. M. Hsieh, and C. M. Rosenthal, Science **303**, 1998 (2004).
[9] R.-B. Wu, R. Long, J. Dominy, T.-S. Ho, and H. Rabitz, Phys. Rev. A **86**, 013405 (2012).
[10] B. Russell, H. Rabitz, and R.-B. Wu, J. Phys. A **50**, 205302 (2017).
[11] C. Ferrie and O. Moussa, Phys. Rev. A **91**, 052306 (2015).
[12] R. S. Judson and H. Rabitz, Phys. Rev. Lett. **68**, 1500 (1992).
[13] C. Brif, R. Chakrabarti, and H. Rabitz, New J. Phys. **12**, 075008 (2010).
[14] J. Roslund, O. M. Shir, T. Bäck, and H. Rabitz, Phys. Rev. A **80**, 043415 (2009).
[15] A. D. Córcoles, J. M. Gambetta, J. M. Chow, J. A. Smolin, M. Ware, J. Strand, B. L. T. Plourde, and M. Steffen, Phys. Rev. A **87**, 030301 (2013).
[16] D. J. Egger and F. K. Wilhelm, Phys. Rev. Lett. **112**, 240503 (2014).
[17] J. Kelly, R. Barends, B. Campbell, Y. Chen, Z. Chen, B. Chiaro, A. Dunsworth, A. G. Fowler, I. C. Hoi, E. Jeffrey *et al.*, Phys. Rev. Lett. **112**, 240504 (2014).
[18] J. Roslund and H. Rabitz, Phys. Rev. A **79**, 053417 (2009).
[19] J. Li, X. Yang, X. Peng, and C.-P. Sun, Phys. Rev. Lett. **118**, 150503 (2017).
[20] D. Lu, K. Li, J. Li, H. Katiyar, A. J. Park, G. Feng, T. Xin, H. Li, G. Long, A. Brodutch *et al.*, npj Quantum Inf. **3**, 45 (2017).
[21] M. A. Rol, C. C. Bultink, T. E. O'Brien, S. R. de Jong, L. S. Theis, X. Fu, F. Luthi, R. F. L. Vermeulen, J. C. de Sterke, A. Bruno *et al.*, Phys. Rev. Applied **7**, 041001 (2017).
[22] M. A. Nielsen and I. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, Cambridge, UK, 2000).
[23] T.-M. Zhang, R.-B. Wu, F.-H. Zhang, T.-J. Tarn, and G.-L. Long, IEEE Trans. Control Syst. Technol. **23**, 2018 (2015).
[24] T. O. Reiss, N. Khaneja, and S. J. Glaser, J. Magn. Reson. **165**, 95 (2003).
[25] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery, *Numerical Recipes: The Art of Scientific Computing* (Cambridge University Press, New York, 2007).
[26] B. Qi, Z. Hou, Y. Wang, D. Dong, H.-S. Zhong, L. Li, G.-Y. Xiang, H. M. Wiseman, C.-F. Li, and G.-C. Guo, npj Quantum Inf. **3**, 19 (2017).
[27] I. A. Pogorelov, G. I. Struchalin, S. S. Straupe, I. V. Radchenko, K. S. Kravtsov, and S. P. Kulik, Phys. Rev. A **95**, 012302 (2017).
[28] S. S. Straupe, JETP Lett. **104**, 510 (2016).