# Deep Learning of Robust and High-Precision Quantum Controls

Re-Bing Wu and Haijin Ding
*Department of Automation, Tsinghua University, Beijing, 100084, P.R. China*
*Center for Quantum Information Science and Technology, BNRist, Beijing, 100084, P.R. China**

Daoyi Dong
*School of Engineering and Information Technology,*
*University of New South Wales, Canberra, ACT 2600, Australia*†

Xiaoting Wang‡
*Institute of Fundamental and Frontier Sciences,*
*University of Electronic Science and Technology of China, Chengdu,610054, China*

(Dated: October 28, 2019)

Robust and high-precision quantum control is extremely important but challenging for scalable quantum computation. In this letter, we show that this hard problem can be translated to a supervised machine learning task by treating the time-ordered controlled quantum evolution as a layer-ordered deep neural network (DNN). The finding of robust quantum controls is then equivalent to training a DNN with high *generalizability*. In this way, powerful DNN tuning skills matured in deep learning (DL) can be employed for the discovery of highly robust and precise quantum controls, which opens up a door through which a large family of learning algorithms can be developed. We exemplify this DL-inspired potential by introducing the commonly used trick of batch-based optimization. The resulting b-GRAPE algorithm is demonstrated to be able to remarkably enhance the control robustness while maintaining high fidelity in the implementation of a three-qubit quantum gate.

PACS numbers: 42.50.Dv, 02.30.Yy

Highly accurate control of quantum hardware is vital for achieving quantum supremacy in the near future [1]. The control design is usually easy with respect to a deterministic model. However, finding a single-shot solution that also tolerates system's uncertainties, e.g., imprecisely identified parameters [2] or time-dependent noises in the Hamiltonian [3], is much harder. Geometrically, the control robustness can be enhanced by minimizing the curvature of control fields on some high-dimensional manifold [4], which leads to various expansion-based methods including the adiabatic method (STIRAP) [5] against control pulse imprecisions, dynamical decoupling (DD) [6–10], difference evolution (SuSSADE) [11, 12] algorithms against environmental noises, and other Taylor-expansion based approaches [13, 14]. Most of these methods (e.g., STIRAP and DD) have been experimentally demonstrated to be effective on suppressing the respective disturbances or noises.

Taking a different and more unified point of view, the design of robust controls can be also thought of as simultaneous optimization of a collection of uncertain quantum systems. Along this route, ensemble- or sampling-based approaches [15–17] are proposed to minimize the average error of the entire collection or multiple samples of quantum systems, respectively. These algorithms are applicable to arbitrary type of uncertainties, including time-independent parameters or time-dependent noises, but in practice they are limited to small-dimensional systems or systems with few uncertainty parameters due to the exponentially increasing computational cost. Even if the computation is affordable, the search for robust controls will be hindered by a large amount of poor solutions over a very complicated landscape.

In this paper, we follow the second route that is capable of dealing with different types of uncertainties, which is usually not feasible by the above expansion-based approaches, especially when several uncertainties coexist in a system. We find that the search for robust quantum controls can be elegantly mapped to the training of a deep neural network (DNN), and enormous powerful techniques developed for the latter in the deep learning (DL) literature can be borrowed to solve our problem.

To see how this works, let us start from the general model of uncertain quantum control systems:

$$\dot{U}(t,\epsilon) \;=\; -iH\left[u(t),\epsilon\right]U(t,\epsilon), \qquad (1)$$

in which the $N\times N$ unitary propagator $U(t,\epsilon)$ is steered from the identity matrix $\mathbb{I}_N$ by the control function $u(t)$. The vector variable $\epsilon\in\mathbb{R}^k$

*Electronic address: rbwu@tsinghua.edu.cn
†Electronic address: daoyidong@gmail.com
‡Electronic address: xiaoting@uestc.edu.cn

denotes the uncertainties in the Hamiltonian that can be some constant but unknown parameters or a time-dependent noise (discretized into a vector of uncertainty parameters). We expect to find a robust control $u(t)$ such that the gate operation $U(T,\epsilon) \in \mathbb{C}^{N\times N}$ equals to some target gate $U_f$ for all possible $\epsilon$. Since such goal is usually not achievable, we can approach it by minimizing the following objective

$$L[u(t)] = \int_{\mathbb{R}^k} \|U(T,\epsilon) - U_f\|^2 P(\epsilon)\mathrm{d}\epsilon, \qquad (2)$$

where $P(\epsilon)$ is the *a priori* probability distribution of the uncertainty parameter. Furthermore, to alleviate the computation burden of the integral (2), we approximate the objective by the average over a finite number of uncertainty parameters sampled from $P(\epsilon)$, say $\mathcal{S} = \{\epsilon_1, \epsilon_2, \cdots\}$, as follows

$$L[u(t), \mathcal{S}] = |S|^{-1} \sum_{\epsilon\in S} \|U(T,\epsilon) - U_f\|^2. \qquad (3)$$

The so-called ensemble-based and sample-based algorithms are subject to the above two cost functions (2) and (3), respectively.

The way we improve the robustness through optimizing (3 is very similar to a supervised machine learning process through minimizing its empirical risk on a set of training samples. As is illustrated in Fig. 1, the quantum control system can be envisioned as a linear neural network (NN) that outputs a unitary propagator from an input uncertainty parameter. Under piecewise-constant controls (e.g., generated from an Arbitrary Waveform Generator), the time-ordered control amplitudes play the role of layer-ordered weight parameters of the equivalent NN, which is said to be deep when there are many control segments. In this way, we may translate the robust quantum control design to a supervised deep learning (DL) task in which the outputs of uncertainty samples are all assigned to the desired quantum gate $U_f$.

The connection to DL provides a new angle for understanding and solving robust quantum control problems. Recall that the goal of DL is not simply fitting a DNN model to the training data, because overly focusing on the empirical risk (3) may lead to the well-known *overfitting* issue. The ultimate goal of DL is to find a generalizable model that also predicts well on new samples outside the training set. Translated to our quantum control problem, the *generalizability* of a learning model [18] to quantum control corresponds exactly to the *robustness* of a control field.

Keeping this in mind, the design of robust quantum controls can be enlightened by vast DL studies on improving the generalizability of a DNN [19], from the following two aspects:

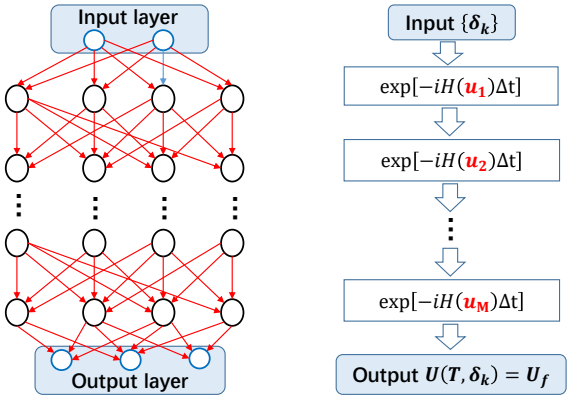


Fig 1: The similarity between a deep neural network (left) and a quantum system under piecewise controls (right). The quantum system outputs a unitary propagator from a given input $\epsilon$, where the time-ordered control amplitudes $u_1, u_2, \cdots, u_M$ play the role of layer-ordered hyper-parameters in a neural network.

(1) *Data augmentation.* One always learns better with more samples. Many DL problems have to learn from limited labeled training samples that are hard to obtain (e.g., diagnosis of professional doctors from medical images). In uncertain quantum control systems, we have *uncountably* many labeled samples [i.e., pairs of $\epsilon \to U_f$ for all admissible $\epsilon \in \mathbb{R}^k$] that can be used for free. However, the existing sampling-based algorithms are only able to exploit a finite number of these samples due to limited computational ability.

(2) *Mini-batch optimization.* The tuning of a DNN with big-data applications is also computationally very expensive, and the central idea for alleviating the burden is that, instead of evaluating the loss with all samples, in each iteration we use only a randomly chosen small batch of samples. In this way, we are still able to explore all samples after sufficiently many iterations. More interestingly, the noisy and thus less stable training dynamics can effectively improve tje *generalizability* by pulling the search away from the weakly attractive poor solutions. This merit has been extensively approved in the practice of DL, and will be also key to our robust quantum control applications.

Concretely, the implementation will be based on the well-known GRAPE algorithm [20] that has been broadly applied for quantum controls. The major difference is that the loss function and its gradient is evaluated with respect to a random batch of samples (whose number $B$ is called the batch size), say

$$\mathcal{S}^{(j)} = \{\epsilon_1^{(j)}, \cdots, \epsilon_B^{(j)}\} \qquad (4)$$

in the $j$-th iteration, as follows:

$$g[u(t),\mathcal{S}^{(j)}] = \frac{\delta L[u(t),\mathcal{S}^{(j)}]}{\delta u(t)} = \frac{1}{B}\sum_{k=1}^{B}\frac{\delta\|U(T,\epsilon_k^{(j)})-U_f\|}{\delta u(t)}. \quad (5)$$

The variational calculation of $\frac{\epsilon\|U(T,\epsilon_k^{(j)})-U_f\|}{\epsilon u(t)}$ is the same as that of GRAPE.

The simplest control updating strategy is to take the "steepest descending" direction along the stochastic gradient (5) with some prescribed learning rate $\alpha_j$:

$$u^{(j+1)}(t) = u^{(j)}(t) - \alpha_j \cdot g[u^{(j)}(t),\mathcal{S}^{(j)}]. \quad (6)$$

Due to the randomness of $\mathcal{S}^{(j)}$, the loss function $L[u(t),\mathcal{S}^{(j)}]$ does not always monotonically decrease along the chosen batch-dependent gradient direction. To stabilize the training dynamics that is often instable under small batch sizes, a very effective method for reducing the variance of the loss function is to introduce a momentum term:

$$u^{(j+1)}(t) = u^{(j)}(t) - \alpha_j \cdot \Big\{\lambda g[u^{(j)}(t),\mathcal{S}^{(j)}] + (1-\lambda) g[u^{(j-1)}(t),\mathcal{S}^{(j-1)}]\Big\}, \quad (7)$$

where the updating direction is dominated by the gradient in previous iterations. The weight parameter $\lambda$ is usually chosen as a small positive real number (e.g., 0.1 or 0.01).

Since the iteration strategies are based on the standard GRAPE algorithm [20] that incorporates randomly selected batches, we term the proposed algorithm as b-GRAPE ("b" stands for "batch"). Correspondingly, the sampling-based algorithm is denoted as s-GRAPE for ("s" for "sampling") [17]. The s-GRAPE algorithm can be taken as a special case of b-GRAPE algorithm when using a fixed batch in all iterations, while GRAPE (for deterministic quantum systems) can be taken as a special case of s-GRAPE when the size of the fixed batch is $B = 1$.

Next, we show by simulation examples how the DL-illuminated b-GRAPE algorithm can effectively harden quantum controls by deeply learning the uncertainties. We choose a three-qubit control system:

$$H(t) = (1+\epsilon_1)\sigma_{1z}\sigma_{2z} + (1+\epsilon_2)\sigma_{2z}\sigma_{3z} + \sum_{k=1}^{3}\left[u_{kx}(t)\sigma_{kx} + u_{ky}(t)\sigma_{ky}\right], \quad (8)$$

where the uncertainty parameters $\epsilon_1$ and $\epsilon_2$ represent the identification errors in the coupling constants (dimensionless after normalization). Each qubit is manipulated by two independent control fields $u_{kx}(t)$ (along $x$-axis) and $u_{ky}(t)$ (along $y$-axis), respectively. The target three-qubit gate $U_f$ is chosen as the Toffoli gate (or controlled-controlled-NOT gate).

To start with, we set the time duration as $T = 10$ and divide each control field evenly into $M = 100$ piecewise constant segments. Assume that the coupling constants may vary by at most $\pm 20\%$, the uncertainty parameters are uniformly sampled from the set $\mathcal{S} = \{(\epsilon_1,\epsilon_2) : |\epsilon_1| \le 0.2, |\epsilon_2| \le 0.2\}$. We test b-GRAPE with three typical batch sizes $B = 1$, 10 and 100, and compare it with s-GRAPE using the same batch sizes and the same initial guess on the control. Because the training stability increases with the batch size, the corresponding learning rates are chosen as $\alpha = 0.002$, 0.02 and 0.2, respectively, to be proportional to the batch size. The resulting training curves, namely the average infidelity evaluated on each batch versus the number of evaluated samples (equal to the batch size times the number of iterations), are shown in Fig. 2 for both b-GRAPE and s-GRAPE algorithms. One can observe the batch-induced noises in the training curves of b-GRAPE, whose variance is large when using small batches. However, a clear trend of decrease is still observable.

Because the average infidelity calculated with small batches may not reflect the actual performance, we reevaluate the performance of the control functions obtained in each iteration by 1000 independently samples drawn from the same probability distribution, as a better approximation of the true average infidelity (2). For s-GRAPE with batch size $B = 1$, the training error can approach the computer machine precision, which is far below the range displayed in the plot, but the testing performance is very poor ($\approx 0.1$). For $B = 10$, the gap between the testing curve (above) and the training curve (below) can be clearly seen, which is an evident indicator of *overfitting* due to the lack of training samples. In contrast, the testing curves of b-GRAPE fit (on average) very well to the training curves, implying that b-GRAPE has better generalizability performance owing to the ability of exploring unlimited number of samples.

The most significant difference, as can be seen with all tested batch sizes, is that b-GRAPE finds much more robust controls than s-GRAPE. For the example of $B = 100$ [see Fig. 2(c)], the generalization gap is almost invisible for s-GRAPE, which implies that the batch size has been sufficiently large to avoid overfitting. Under this circumstance, b-GRAPE still performs much better than s-GRAPE, because the batch-induced noises can steer the search away from poor solutions. The best result is achieved when using the small-

est batch size $B = 1$ [see Fig. 2(a)], under which the average infidelity can be reduced to be below 0.001 (lower than the error correction threshold). The contrast again testifies for the active role of batch-induced noises, which is the strongest when $B = 1$, on guiding the search toward more robust solutions.

To manifest the degree of robustness enhancement, we compare control fields obtained by GRAPE ($B = 1$, fixed batch), s-GRAPE ($B = 100$, fixed batch) and b-GRAPE ($B = 1$, random batch) algorithms via their 3-D robustness landscapes (i.e., the infidelity versus the two uncertainty parameters, see Fig. 3), on which the robustness can be quantified by the area enclosed by the level set at some threshold value (say 0.001 in the figure). The control obtained by GRAPE from a deterministic model achieves extremely high precision at the chosen sample $\epsilon_1 = \epsilon_2 = 0$, but the sharp minimum indicates that the control is very sensitive to the uncertainty and hence is non-robust. By contrast, the landscapes corresponding to s-GRAPE and b-GRAPE algorithm cases are much flatter. The controls sacrifice the precision near the sample $\epsilon_1 = \epsilon_2 = 0$, but maintain high precision in a much broader region. Quantitatively, s-GRAPE and b-GRAPE enhance the robustness by about 4 times and 41 times, respectively. The level set at 0.001 achieved by b-GRAPE almost fills the full $0.2 \times 0.2$ square from which the training samples are drawn.

We also test the performance of b-GRAPE with less available control resources, i.e., using shorter time duration ($T = 5$) and less number ($M = 50$) of control segments. The simulations are all stopped after exploring one million samples. Not surprisingly, the ability of robustness enhancement decreases when the control is more limited, as can be seen from examples shown in Fig. 4. However, the simulations consistently demonstrate that the superiority of b-GRAPE (bold black) over s-GRAPE (dotted red) in all cases. In fact, b-GRAPE should be more advantageous under such circumstances because otherwise the search will be more easily trapped by local optima.

The simulations also includes an exceptional example shown in Fig. 4($b_1$), whose robustness achieved by b-GRAPE is supposed to be stronger than those in Fig. 4($a_1$) for the control time is longer and Figs. 4($b_2$-$b_3$) for the the batch size is smaller. This poor solution results from an instable training process, during which the average infidelity over training batches rises up after about exploring about 800 thousands of samples. The level set at 0.001 (in blue) corresponding to the best solution before losing stability is also depicted in Fig. 4($b_1$), which is disconnected due to the co-

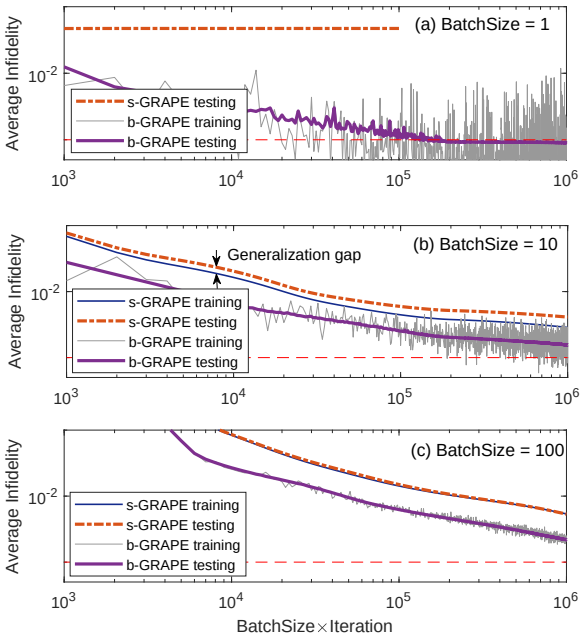


Fig 2: The training and testing curves with b-GRAPE and s-GRAPE algorithms, where the controls are over the time duration $[0, 10]$ and are divided into 100 segments. Under all selected batch sizes (a) $B = 1$, (b) $B = 10$ and (c) $B = 100$, b-GRAPE outperforms s-GRAPE.

existence of two minima like in Fig. 3. Nevertheless, the training process is not always instable, and very robust solutions can be obtained after restarting the stochastic b-GRAPE optimization, reselecting the initial guess or decrease the learning rates.

To conclude, we proposed b-GRAPE, a deep-learning illuminated algorithm, for efficiently discovering highly robust quantum controls in high-precision regime. The algorithm can be easily implemented by randomizing the renowned GRAPE algorithm with batches of samples, and numerical simulations demonstrate its effectiveness owing to the endowed ability of exploring uncountably many available uncertainty samples, and the ability of escaping poor optima driven by the batch-induced randomness. Moreover, this batch-based algorithm can be naturally paralleled. Although the theoretically best performance is usually achieved when $B = 1$, in practice we can use adequately larger batches to increase the computational efficiency as well as the algorithmic stability.

It should be noted that some recent studies on robust quantum control design were also inspired by supervised machine learning [21] or reinforcement learning [22–24]. These works took a different perspective in that external DNNs are introduced to remodel and learning the uncertain quan-

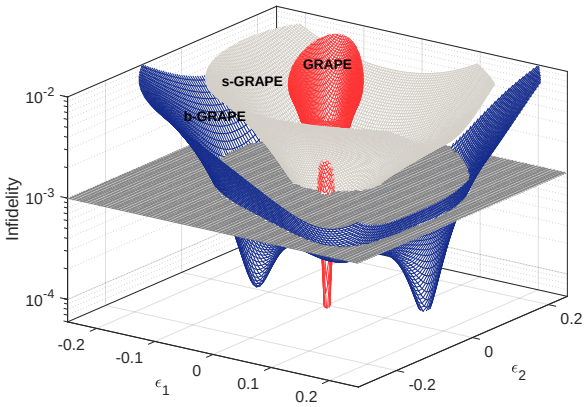


Fig 3: The robustness control landscape defined as the gate infidelity versus two uncertainty parameters under controls optimized with GRAPE (red), s-GRAPE (white) using 100 samples and b-GRAPE with batch size $B = 1$ (blue). The b-GRAPE obtains a much more robust control than the other two schemes, where the robustness is quantified by the area enclosed by the level set at 0.001.

tum control dynamics, while the quantum system is taken as a black-box data generator for training the DNNs. Our approach does not require any additional learning structures because the quantum system itself is a natural "DNN" that can be taught to learn. Our algorithm is also distinct from SPSA (Simultaneous Perturbation Stochastic Algorithm), another type of stochastic gradient algorithm, which calculates the gradient by randomizing the projected direction instead of the samples. The latter had been proposed for online model-free learning of robust quantum control and tomography [25], and we believe that it can be combined with b-GRAPE for broader applications.

In our numerical tests, no evident traps (i.e., local optima that are far from global optima) were encountered, which exhibit nice control landscapes that have been observed in both quantum control [26–28] and deep learning [29, 30] studies. Under proper assumptions, both landscapes can be proved to be devoid of traps, and the landscape of deep learning is often hindered by abundantly existing landscape saddles, and these facts are supported by vast and continually growing set of successful tests. We conjecture that the control landscape of uncertain quantum systems has similar structures, and a further study on this topic will be very important to a full understanding of robust quantum control problems.

This work is only the start of a potentially large family of robust control design algorithms, because tremendously many more powerful skills on tuning DNNs can be transplanted here, e.g., Adam, AdaGrad, REProp [19] or Newton-like stochastic gradient algorithms, namely Newton sketch [31]). As can be seen in our simulations, there is still much room for the control robustness to be enhanced, e.g., by fine tuning the learning rates, increasing the number of iterations and introducing more advanced tuning techniques. We expect to such DL-inspired algorithms can experimentally applied for high-quality control of quantum information processing hardware in the near futere.

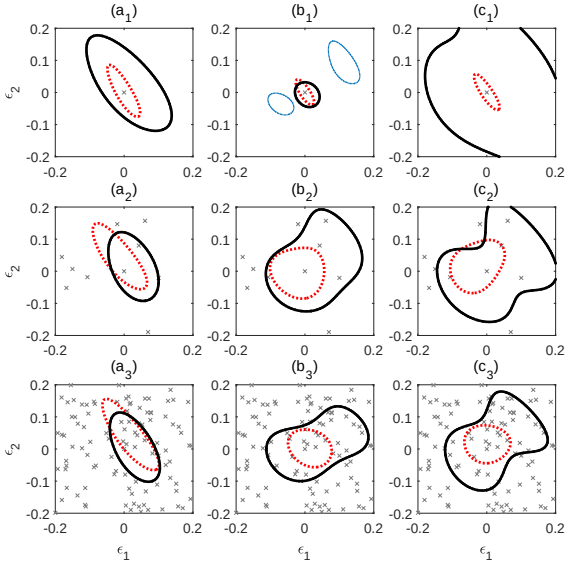


Fig 4: The robustness (quantified by the area enclosed by the level set at 0.001) of control fields obtained by b-GRAPE (black solid) and s-GRAPE (red dotted) algorithms. The time duration $T$ and number of control segments $M$ are, respectively, ($a_1$-$a_3$) $T = 5$ and $M = 50$; ($b_1$-$b_3$) $T = 10$ and $M = 50$; ($c_1$-$c_3$) $T = 10$ and $M = 100$. The corresponding batch sizes are $B = 1$ (1st row), $B = 10$ (2nd row) and $B = 100$ (3rd row)


[1] A. W. Harrow and A. Montanaro, Nature **549**, 203 (2017).

[2] M. Dahleh, A. P. Peirce, and H. Rabitz, Phys. Rev. A **42**, 1065 (1990).

[3] M. Demiralp and H. Rabitz, Physical Review A **57**, 2420 (1998).

[4] D. Hocker, C. Brif, M. D. Grace, A. Donovan, T.-S. Ho, K. M. Tibbetts, R. Wu, and H. Rabitz, Physical Review A **90**, 062309 (2014).

[5] N. V. Vitanov, A. A. Rangelov, B. W. Shore,

and K. Bergmann, Reviews of Modern Physics **89**, 015006 (2017).
[6] L. Viola, E. Knill, and S. Lloyd, Physical Review Letters **82**, 2417 (1999).
[7] M. J. Biercuk, H. Uys, A. P. VanDevender, N. Shiga, W. M. Itano, and J. J. Bollinger, Nature **458**, 996 (2009).
[8] A. M. Souza, G. A. Alvarez, and D. Suter, Physical Review Letters **106**, 240501 (2011).
[9] T. J. Green, J. Sastrawan, H. Uys, and M. J. Biercuk, New Journal of Physics **15**, 095004 (2013).
[10] L. F. Santos and L. Viola, New Journal of Physics **10**, 083009 (2008).
[11] E. Zahedinejad, J. Ghosh, and B. C. Sanders, Physical Review Letters **114**, 200502 (2015).
[12] E. Zahedinejad, J. Ghosh, and B. C. Sanders, Physical Review Applied **6**, 54005 (2016).
[13] D. Daems, A. Ruschhaupt, D. Sugny, and S. Guérin, Phys. Rev. Lett. **111**, 050404 (2013).
[14] C.-H. Huang and H.-S. Goan, Phys. Rev. A **95**, 062325 (2017).
[15] J. S. Li and N. Khaneja, Physical Review A **73**, 030302 (2006).
[16] J. Ruths and J.-S. Li, Ieee Transactions on Automatic Control **57**, 2021 (2012).
[17] C. Chen, D. Dong, R. Long, I. R. Petersen, and H. A. Rabitz, Physical Review A **89**, 023402 (2014).
[18] H. Xu and S. Mannor, Machine Learning **86**, 391 (2012).
[19] L. Bottou, F. E. Curtis, and J. Nocedal, Siam Review **60**, 223 (2018).
[20] N. Khaneja, T. Reiss, C. Kehlet, T. Schulte-Herbruggen, and S. J. Glaser, J. Magn. Reson. **172**, 296 (2005).
[21] X.-C. Yang, M.-H. Yung, and X. Wang, Phys. Rev. A **97**, 042324 (2018).
[22] M. Bukov, A. G. R. Day, D. Sels, P. Weinberg, A. Polkovnikov, and P. Mehta, Phys. Rev. X **8**, 031086 (2018).
[23] J. M. Arrazola, T. R. Bromley, J. Izaac, C. R. Myers, K. Brádler, and N. Killoran, ArXiv e-prints (2018), 1807.10781.
[24] M. Yuezhen Niu, S. Boixo, V. Smelyanskiy, and H. Neven, ArXiv e-prints (2018), 1803.01857.
[25] C. Ferrie and O. Moussa, Phys. Rev. A **91**, 052306 (2015).
[26] H. A. Rabitz, M. M. Hsieh, and C. M. Rosenthal, Science **303**, 1998 (2004).
[27] H. Rabitz, M. Hsieh, and C. Rosenthal, Physical Review A **72**, 052337 (2005).
[28] R. B. Wu, H. Rabitz, and M. Hsieh, Journal of Physics A-mathematical and Theoretical **41**, 015006 (2008).
[29] K. Kawaguchi, ArXiv e-prints (2016), 1605.07110.
[30] P. Zhou and J. Feng, ArXiv e-prints (2017), 1705.07038.
[31] M. Pilanci and M. Wainwright, SIAM Journal on Optimization **27**, 205 (2017).