Autotuning of Double-Dot Devices In Situ with Machine Learning

Justyna P. Zwolak, Thomas McJunkin, Sandesh S. Kalantre, J.P. Dodson, E.R. MacQuarrie, D.E. Savage, M.G. Lagally, S.N. Coppersmith, Mark A. Eriksson, and Jacob M. Taylor

Phys. Rev. Applied **13**, 034075 — Published 31 March 2020

DOI: 10.1103/PhysRevApplied.13.034075

Auto-tuning of double dot devices *in situ* with machine learning

Justyna P. Zwolak,[1,*] Thomas McJunkin,[2,†] Sandesh S. Kalantre,[3,4] J. P. Dodson,[2] E. R. MacQuarrie,[2] D. E. Savage,[5] M. G. Lagally,[5] S. N. Coppersmith,[2,6] Mark A. Eriksson,[2] and Jacob M. Taylor[1,3,4]

[1]*National Institute of Standards and Technology, Gaithersburg, Maryland 20899, USA*
[2]*Department of Physics, University of Wisconsin-Madison, Madison, Wisconsin 53706, USA*
[3]*Joint Quantum Institute, University of Maryland, College Park, Maryland 20742, USA*
[4]*Joint Center for Quantum Information and Computer Science, University of Maryland, College Park, Maryland 20742, USA*
[5]*Department of Materials Science and Engineering, University of Wisconsin-Madison, Madison, Wisconsin 53706, USA*
[6]*School of Physics, The University of New South Wales, Sydney, New South Wales, Australia*

(Dated: January 21, 2020)

The current practice of manually tuning quantum dots (QDs) for qubit operation is a relatively time-consuming procedure inherently impractical for scaling up and applications. In this work, we report on the *in situ* implementation of a recently proposed auto-tuning protocol that combines machine learning (ML) with an optimization routine to navigate the parameter space. In particular, we show that a ML algorithm trained using exclusively simulated data to quantitatively classify the state of double QD device can be used to replace human heuristics in tuning of gate voltages in real devices. We demonstrate active feedback of a functional double dot device operated at millikelvin temperatures and discuss success rates as a function of initial conditions and device performance. Modifications to the training network, fitness function, and optimizer are discussed as a path towards further improvement in the success rate when starting both near and far detuned from the target double dot range.

I. INTRODUCTION

Arrays of quantum dots (QDs) are one of many candidate systems to realize qubits—the fundamental building blocks of quantum computers—and to provide a platform for quantum computing [1–3]. Due to the ease of control of the relevant parameters [4–7], fast measurement of the spin and charge states [8], long decoherence times [9–11], and recent demonstration of two qubit gates and algorithms [12–14], QDs are gaining popularity as candidate building blocks for solid-state quantum devices. In semiconductor quantum computing, devices now have tens of individual gate voltages that must be carefully set to isolate the system to the single electron regime and to realize good qubit performance. At the same time, even tuning a double QD constitutes a nontrivial task, with each dot being controlled by at least three metallic gates, each of which influences the number of electrons in the dot, the tunnel coupling to the adjacent lead, and the interdot tunnel coupling. The background potential energy, which is disordered by defects and variations in the local composition of the heterostructure, further impedes this process. In order to reach a stable, few electron configuration, current experiments set the input voltages heuristically. However, such an approach does not scale well with growing array sizes, is prone to random errors, and may result in only an acceptable rather than an optimal state. Moreover, with an increasing number of QD qubits, the relevant parameter space grows exponentially, making heuristic control even more challenging.

Given the recent progress in the physical construction of larger arrays of quantum dots in both one and two dimensions [15, 16], it is imperative to have a reliable automated protocol to find a stable, desirable electron configuration in the dot array, i.e., to automate finding a set of voltages that yield the desired confinement regions (dots) at the intended positions and with the correct number of electrons and couplings, and to do it efficiently. There have been a number of recent proposals on how to achieve these tasks, including computer-supported, algorithmic gate voltage control and pattern matching for tuning [17–21] and machine learning guided protocol aimed at reducing the total number of measurements [22]. However, while these tuning approaches to a lesser or greater extent eliminate the need for human intervention, they are tailored to a particular device's design and need to be adjusted if used on a different one. Moreover, most of these approaches focus on fine-tuning to the single-electron regime, assuming some level of knowledge about the parameter ranges that lead to a well-controlled qubit system.

Typically, the process of tuning QD devices into qubits involves identifying the global state of the device (e.g., single dot or double dot) from a series of measurements, followed by an adjustment of parameters (gate voltages) based on the observed outcomes. The classification of outcomes can be determined by a trained researcher, identifying the location of QDs based on the relative action of gates and the assembly of multiple QDs based on the relative capacitive shifts. In recent years, machine learning (ML) algorithms, and specifically convolutional

* jpzwolak@nist.gov
† tmcjunkin@wisc.edu

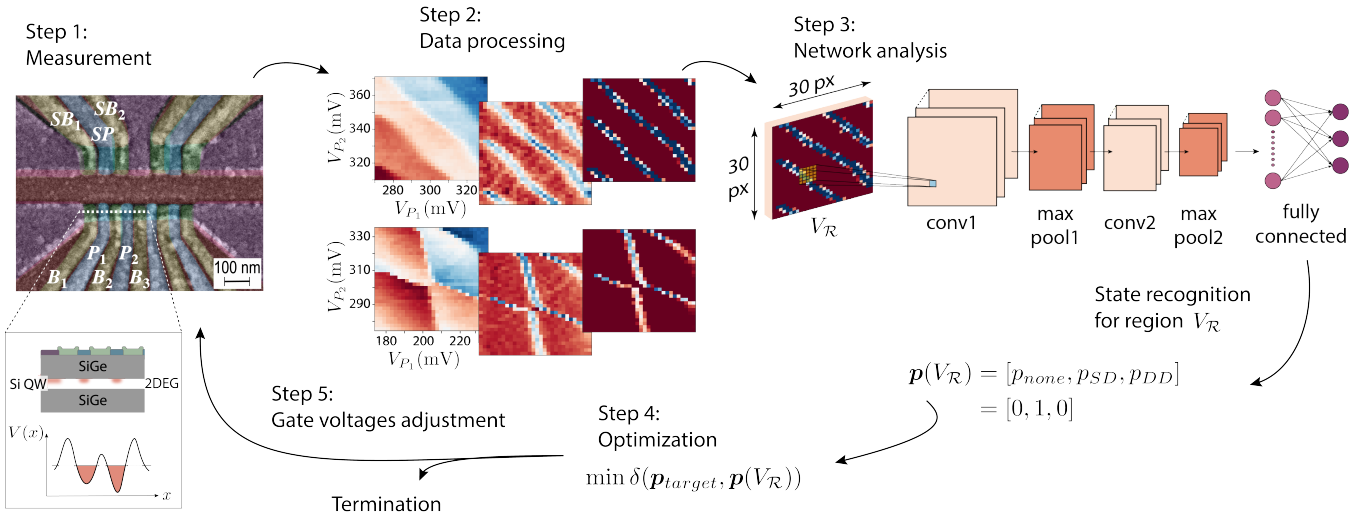

FIG. 1. Visualization of the auto-tuning loop. In Step 1, we show a false-color scanning electron micrograph of a Si/SiGe quadruple dot device identical to the one measured. The double dot used in the experiment is highlighted by the inset, which shows a cross-section through the device along the dashed white line and a schematic of the electric potential of a tuned double dot. B_i $(i = 1, 2, 3)$ and P_j $(j = 1, 2)$ are the barrier and plunger gates, respectively, used to form dots, while SB_1, SB_2, and SP are gates (two barriers and a plunger, respectively) used to control the sensing dot. In Step 2, to assure compatibility with the CNN, the raw data is processed and (if necessary) downsized to (30×30) pixel size. The processed image $V_{\mathcal{R}}$ is analyzed by the CNN (Step 3), resulting in a probability vector $\boldsymbol{p}(V_{\mathcal{R}})$ quantifying the current state of the device. In the optimization phase (Step 4), the algorithm decides whether the state is sufficiently close to the desired one (termination) or whether additional tuning steps are necessary. If the latter, the optimizer returns the position of the consecutive scan (Step 5).

neural networks (CNNs), have emerged as a "go to" technique for automated image classification, giving reliable output when trained on a representative and comprehensive dataset [23]. Recently, Kalantre *et al.* have proposed a new paradigm for fully automated experimental device control – *QFlow* – that combines CNNs with optimization techniques to establish a closed-loop control system [24]. Here, we report on the performance of this auto-tuning protocol when implemented *in situ* on an active quantum dot device to tune from a single dot to a double dot regime. We also discuss further modifications to this protocol to improve overall performance.

The paper is organized as follows: In Section II, we describe the experimental setup. The characteristics of the quantum dot chip used in the experiment are described in Section II A. An overview of the machine learning and optimization techniques implemented in the auto-tuning protocol is presented in Section II B and Section II C, respectively. The *in situ* performance of the auto-tuner is discussed in Section III and the "off-line" analysis in Section IV. We conclude with a discussion of the potential modifications to further improve the proposed auto-tuning technique in Section V.

II. EXPERIMENTAL SETUP

We define "auto-tuning" as a process of finding a range of gate voltages where the device is in a particular "global configuration" (i.e., no dot, single dot, or double dot regime). The main steps of the experimental implementation of the auto-tuner are presented in Fig. 1, with each step discussed in detail in the following sections.

Step 0: Preparation. Before the machine learning systems are engaged, the device is cooled down and gates are manually checked for response and pinch-off voltages. Furthermore, the charge sensor and the barrier gates are also tuned using traditional techniques.

Step 1: Measurement. A 2D measurement of the charge sensor response over a fixed range of gate voltages. The position for the initial measurement (given as a center and a size of the scan in mV) is provided by a user.

Step 2: Data processing. Re-sizing of the measured 2D scan $V_{\mathcal{R}}$ and filtering of the noise (if necessary) to assure compatibility with the neural network.

Step 3: Network analysis. Analysis of the processed data. The CNN identifies the state of the device for $V_{\mathcal{R}}$ and returns a probability vector $\boldsymbol{p}(V_{\mathcal{R}})$, see Eq. (1).

Step 4: Optimization. An optimization of the fitness function $\delta(\boldsymbol{p}_{target}, \boldsymbol{p}(V_{\mathcal{R}}))$, given in Eq. (2), resulting either in a position of the consecutive 2D scan or decision to terminate the auto-tuning.

Step 5: Gate voltages adjustment. An adjustment of the gate voltages as suggested by the optimizer. The position of the consecutive scan is given as a center of the scan (in mV).

The Preparation Step results in a range of acceptable voltages for gates, which allows "sandboxing" by limit-

ing the two plunger voltages controlled by auto-tuning protocol within these ranges to prevent device damage, as well as in establishing the appropriate voltage level at which the barrier gates are fixed throughout the test runs (pre-calibration). The charge sensing dot is also tuned manually at this stage. The sandbox also helps define the size of the regions used for state recognition. Proper scaling of the measurement scans is crucial for meaningful network analysis: Scans that are too small may not contain enough features necessary for state classification while scans that are too large may result in probability vectors that are not useful in the optimization phase.

Steps one through five are repeated until the desired global state is reached. In other words, we formulate the auto-tuning as an optimization problem over the state of the device in the space of gate voltages, where the function to be optimized is a fitness function δ between probability vectors of the current and the desired measurement outcomes. The auto-tuning is considered successful if the optimizer converges to a voltage range that gives the expected dot configuration.

A. Device layout and characteristics

QDs are defined by electrostatically confining electrons using voltages on metallic gates applied above a two-dimensional electron gas (2DEG) present at the interface of a semiconductor heterostructure. Realization of good qubit performance is achieved via precise electrostatic confinement, band-gap engineering, and dynamically adjusted voltages on nearby electrical gates. A false-color scanning electron micrograph of a Si/SiGe quadruple dot device identical to the one measured is shown in Fig. 1, Step 1. The device is an overlapping, accumulation style design [25] consisting of three layers of aluminum surface gates, electrically isolated from the heterostructure surface by a deposited aluminum oxide. The layers are isolated from each other by the self-oxidation of the aluminum. The inset in Fig. 1 shows a schematic cross-section of the device showing where QDs are expected to form and a modeled potential profile along a one-dimensional (1D) channel formed in the 2DEG. The 2DEG, with an electron mobility of $40\,000$ cm^2/(Vs) at 4.0×10^{11} cm^{-2} as measured in a Hall bar, is formed approximately 33 nm below the surface at the upper interface of the silicon quantum well. Applying appropriate voltages to the gates defines the QDs by selectively accumulating and depleting regions within the 2DEG. In particular, depletion 'screening' gates (shown in red in Fig. 1) are used to define a 1D transport channel in the 2DEG; reservoir gates (shown in purple in Fig. 1) accumulate electrons into leads with stable chemical potential; plunger gates (shown in blue and labeled P_j, $j = 1, 2$, in Fig. 1) accumulate electrons into quantum dots and shift the chemical potential in the dots relative to the chemical potential of the leads; finally, barrier gates (shown in green and labeled B_i, $i = 1, 2, 3$, in Fig. 1) separate the defined quantum dots and control the tunnel rates between dots and to the leads. In other words, the choice of gate voltages determines the number of dots, their position, their coupling, and the number of electrons present in each dot. Across the central screening gate, opposing the main channel of four linear dots, larger quantum dots are formed to act as sensitive charge sensors capable of detecting single electron transitions of the main channel quantum dots. The measurements are taken in a dilution refrigerator with a base temperature < 50 mK and in the absence of an applied magnetic field.

B. Quantitative classification

To automate the tuning process and eliminate the need for human intervention, we incorporate machine learning techniques into the software controlling the experimental apparatus. In particular, we use a pre-trained CNN to determine the current global state of the device. To prepare the CNN, we rely on a dataset of 1001 quantum dot devices generated using a modified Thomas-Fermi approximation to model a set of reference semiconductor systems comprising of a quasi-1D nanowire with a series of depletion gates whose voltages determine the number of dots, the charges on each of those dots, and the conductance through the wire [26, 27]. The dataset has been constructed to be agnostic about the details of a particular geometry and material platform used for fabricating dots. To reflect the minimum qualitative features across a wide range of devices, a number of parameters were varied between simulations, such as the device geometry, gate positions, lever arm, and screening length, to name a few. The idea behind varying the device parameters when generating training dataset was to enable using the same pre-trained network on different experimental devices.

The synthetic dataset contains full size simulated 2D measurements of the charge sensor readout and the state labels at each point as functions of plunger gate voltages (V_{P_1}, V_{P_2}) (at a pixel level). For training purposes, we generated an assembly of $10\,010$ random charge sensor measurement realizations (10 samples per full size scan), with charge sensor response data stored as (30×30) pixel maps from the space of plunger gates (see the right column in Fig. A.6 for examples of simulated single and double dot regions, respectively). The labels for each measurement are assigned based on the probability of each state within a given realization, i.e., based on the fraction of pixels in each of the three possible states:

$$\begin{aligned} \boldsymbol{p}(V_{\mathcal{R}}) &= [p_{none},\, p_{SD},\, p_{DD}] \\ &= \left[\frac{N - (|SD| + |DD|)}{N}, \frac{|SD|}{N}, \frac{|DD|}{N}\right] \end{aligned} \tag{1}$$

where $|SD|$ and $|DD|$ are the numbers of pixels with a single dot and a double dot state label, respectively, and N is the size of the image $V_{\mathcal{R}}$ in pixels. As such, $\boldsymbol{p}(V_{\mathcal{R}})$

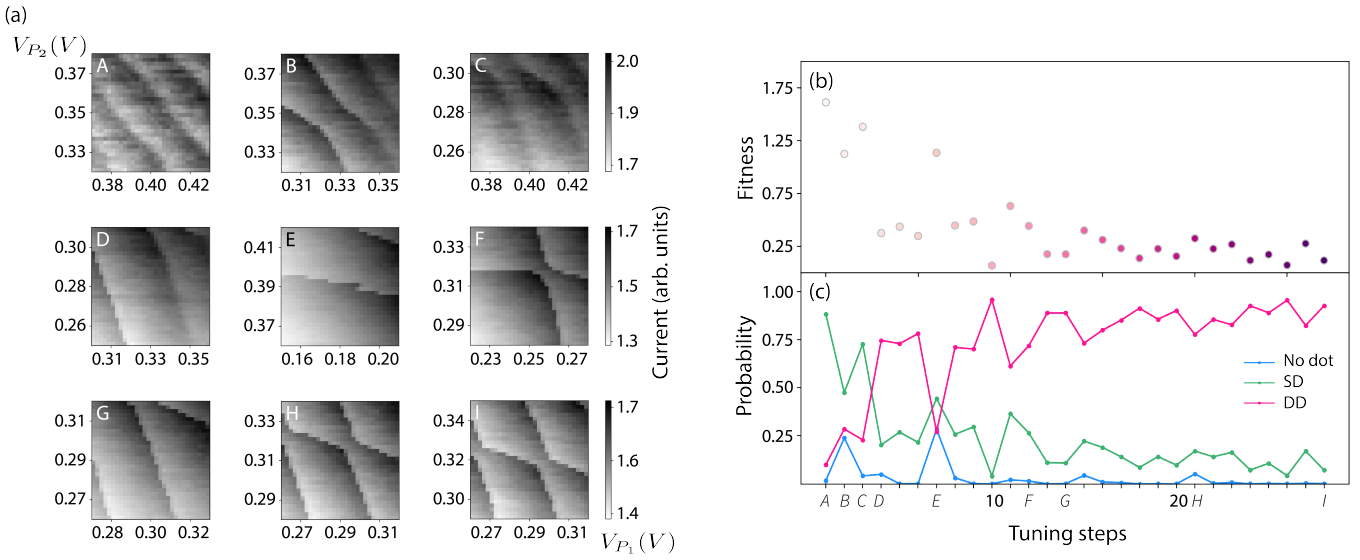

FIG. 2. A sample run of the auto-tuning protocol. (a) The measured raw scans in the space of plunger gates (V_{P_1}, V_{P_2}) show data available to the auto-tuning protocol at a given time. (b) The change of the fitness value as a function of time. (c) The change in probability of each state over time as returned by the CNN. See Fig. 3 for an overview of the tuning path in the space of plunger gates on a larger scan measured once the auto-tuning tests were completed.

can be thought of as a probability vector that a given measurement captures each of the possible states (i.e., no dot, single dot, double dot). The resulting probability vector for a given region $V_{\mathcal{R}}$, $\boldsymbol{p}(V_{\mathcal{R}})$, is an implicit function of the plunger gate voltages defining $V_{\mathcal{R}}$. It is important to note that, while CNNs are traditionally used to simply classify images into a number of predefined global classes (which can be thought of as a *qualitative classification*), we use the raw probability vectors returned by the CNN (i.e., *quantitative classification*).

The CNN architecture consists of two convolutional layers (each followed by a pooling layer) and four fully connected layers with 1024, 512, 256, and 3 units, respectively. The convolutional and pooling layers are used to reduce the size of the feature maps while extracting the most important characteristics of the data. The fully connected layers, on the other hand, allow for non-linear combinations of these characteristics and classification of the data. We use the Adam optimizer [28] with a learning rate $\eta = 0.001$, 5000 steps per training, and a batch size of 50. The accuracy of the network on the test set is 97.7 %.

C. Optimization and auto-tuning

The optimization step of the auto-tuning process (Step 4 in Fig. 1) involves minimization of a fitness function that quantifies how close a probability vector returned by the CNN, $\boldsymbol{p}(V_{\mathcal{R}})$, is to the desired vector, $\boldsymbol{p}_{target}$. We use a modified version of the original fitness function proposed in Ref. [24] to include a penalty for tuning to single dot and no dot regions:

$$\delta(\boldsymbol{p}_{target}, \boldsymbol{p}(V_{\mathcal{R}})) = \|\boldsymbol{p}_{target} - \boldsymbol{p}(V_{\mathcal{R}})\|_2 + \gamma(V_{\mathcal{R}}), \quad (2)$$

where $\|\cdot\|_2$ is the L^2-norm and the penalty function γ is defined as:

$$\gamma(V_{\mathcal{R}}) = \alpha g(p_{none}) + \beta g(p_{SD}), \quad (3)$$

where $g(x)$ is the arctangent shifted and scaled to assure that the penalty is non-negative (i.e., $g(x) \geq 0$) and that the increase in penalty is more significant once a region is classified as predominantly non-double dot (i.e., the inflection point is at $x = 0.5$). Parameters α and β are used to weight penalties coming from no-dot and single dot, respectively.

For optimization, we use the NelderMead method [29, 30] implemented in Python [31]. The Nelder-Mead algorithm works to find a minimum of an objective function by evaluating it at initial simplex points—a triangle in the case of the 2D gate space in this work. Depending on the values of the objective function at the simplex points, the subsequent points are selected to move the overall simplex towards the function minimum. In our case, the initial simplex is defined by the fitness value of the starting region $V_{\mathcal{R}}$ and two additional regions obtained by lowering the voltage on each of the plungers one at a time by 75 mV.

III. AUTO-TUNING THE DEVICE *IN SITU*

To evaluate the auto-tuner in an experimental setup, a Si/SiGe quadruple quantum dot device (see Fig. 1, Step

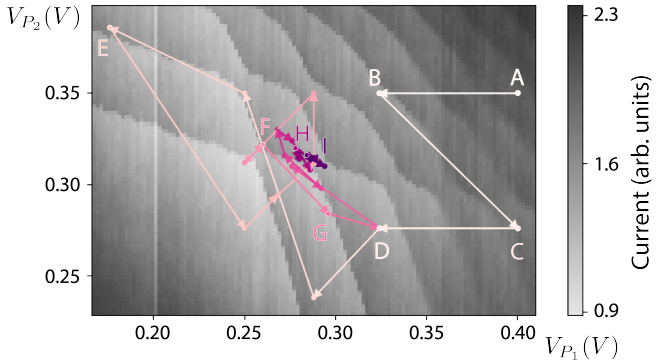

FIG. 3. An overview of a sample run of the auto-tuning protocol in the space of plunger gates (V_{P_1}, V_{P_2}). The arrows and the intensity of the color indicate the progress of the auto-tuner. The palette correspond to colors used in the fitness function plot in Fig. 2.

1) was pre-calibrated into an operational mode, with one double quantum dot and one sensing dot active. The evaluation was carried on in the there main phases: In the first phase, we developed a communication protocol between the auto-tuning software [32] and the software used to control the experimental apparatus [33]. In the process, we collected 83 measurement scans that were then used to refine the filtering protocol used in Step 2 (see the middle column in Fig. A.6). These scans were also used to test the classification accuracy for the neural network.

In the second phase, we evaluated the performance of the trained network on hand-labeled experimental data. The dataset includes (30×30) mV scans with 1 mV/pixel and (60×60) mV with 2 mV/pixel. Prior to analysis, all scans were flattened with an automated filtering function to assure compatibility with the neural network (see the left column in Fig. A.6). The accuracy of the trained network in distinguishing between single dot, double dot, and no dot patterns is 81.9 %.

In the third phase, we performed a series of trial runs of the auto-tuning algorithm in the (V_{P_1}, V_{P_2}) plunger space as shown in Fig. 2. To prevent tuning to voltages outside of the device tolerance regime, we sandbox the tuner by limiting the allowed plunger values to between 0 and 600 mV. Attempts to perform measurement outside of these boundaries during a tuning run are blocked and a fixed value of 2 (i.e., maximum fit value) is assigned to the fitness function.

We initialized 45 auto-tuning runs, out of which 7 were terminated by the user due to technical problems (e.g., stability of the sensor). Of the remaining 38 completed runs, in 13 cases the scans collected at an early stage of the tuning process were found to be incompatible with the CNN. In particular, while there are three possible realizations of the single dot state (coupled strongly to the left plunger, the right plunger, or equally coupled forming a "central dot"), the training dataset included predominantly realizations of the "central dot" state. As a result, whenever the single left or right plunger dot was measured, the scan was labeled incorrectly. When a sequence of consecutive "single plunger dot" scans was used in the optimization step, the optimizer mis-identified the scans as double dot and failed to tune away from this region. These runs were removed from further analysis as with the incorrect labels, the auto-tuner each time terminated in a region classified as double dot (i.e., a success from ML perspective) which in reality was an single dot (i.e., a failure for practical purposes). We discuss the performance of the auto-tuner based on the remaining 25 runs.

TABLE I. Summary of the performance for the experimental test runs ($N_{tot} = 14$). N_{exp} denotes the number of experimental runs initiated at position (V_{P_1}, V_{P_2}) (mV), N_{suc} indicates the number of successful experimental runs, and $P_{\Delta=75}$ (%), $P_{\Delta=100}$ (%), and $P_{\Delta=f(\delta_0)}$ (%) are the success rates for the 81 test runs with optimization parameters resembling the experimental configuration (fixed simplex size $\Delta = 75$ mV), with the initial simplex size increased to 100 mV, and with initial simplex size dynamically adjusted based on the fitness value of the first scan, respectively. All test runs were performed using the new neural network.

(V_{P_1}, V_{P_2})	N_{exp}	N_{suc}	$P_{\Delta=75}$	$P_{\Delta=100}$	$P_{\Delta=f(\delta_0)}$
(250,400)	1	1	85.2	100.0	93.8
(350,400)	6	6	74.1	95.1	95.1
(350,415)	1	0	75.3	86.4	96.3
(350,425)	1	1	55.6	86.4	85.2
(350,450)	3	2	3.7	18.5	34.6
(400,350)	1	1	4.9	69.1	93.8
(450,350)	1	1	17.3	1.2	23.5

While tuning, it has been observed that the auto-tuner tended to fail when initiated further away from the target double dot region. An inspection of the test runs confirms that whenever both plungers were set at or above 375 mV, the tuner became stuck in the plateau area of the fitness function and did not reach the target area (with two exceptions). Out of the 25 completed runs, 14 were initiated with at least one plunger set below 375 mV. Out of these, 2 cases failed, both due to instability of the charge sensor resulting in unusually noisy data that was incorrectly label by the CNN and thus lead to an inconsistent gradient direction. The overall success rate here was 85.7 % (see Table I for a summary of the performance for each initial point from this class). When both plungers were set at or above 375 mV, only two out of 11 runs were successful (18.2 %), with all failing cases resulting from "flatness" of the fit function (see Fig. C.7 for a visualization of the fitness function over a large range of voltages in the space of plunger gates (V_{P_1}, V_{P_2})).

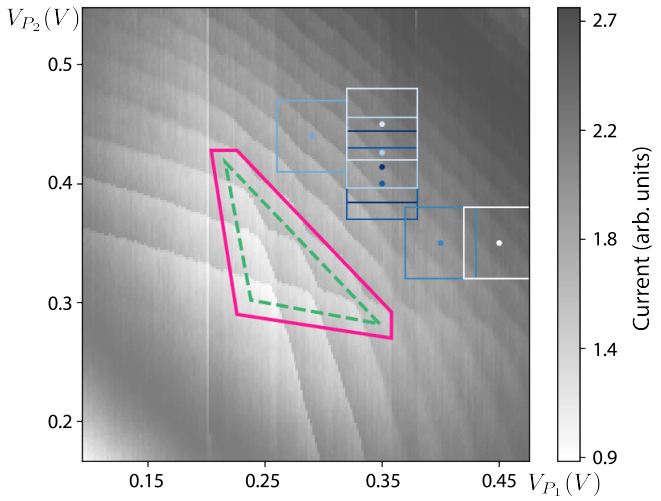

FIG. 4. Visualization of the "ideal" (marked with dashed green triangle) and the "sufficiently close" (marked with solid magenta diamond) regions used to determine the success rate for the off-line tuning. All considered initial regions listed in Table I are marked with squares. The intensity of the colors correspond to the success rate when using dynamic simplex (darker color denotes higher success rate).

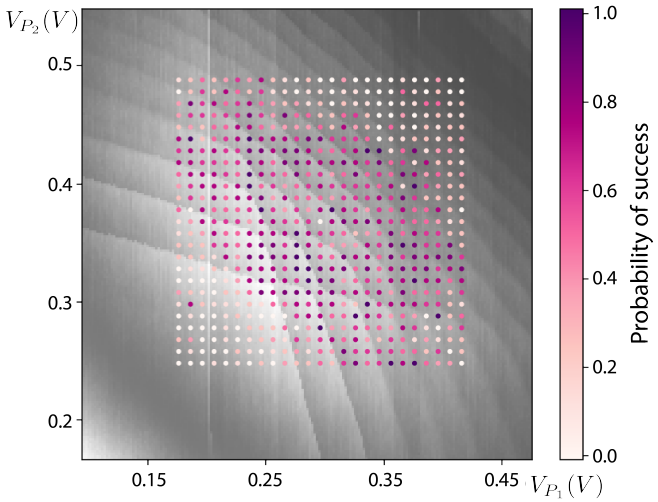

FIG. 5. A heat map of the probability of success when tuning off-line over a set of $N = 4$ pre-measured devices. The intensity of the colors corresponds to the success rate with darker color denoting higher success rate.

IV. "OFF-LINE" TUNING

Tuning "off-line"– tuning within a pre-measured scan for a large range of gate voltages that captures all possible state configurations – allows for the study of how the various parameters of the optimizer impact the functioning of the auto-tuner and the further investigation of the reliability of the tuning process while not taking experimental time. The scan we use spans 125 mV to 525 mV for plunger P_1 and 150 mV to 550 mV for P_2, measured in 2 mV/pixel resolution.

The deterministic nature of the CNN classification (i.e., assigning a fixed probability to a given scan) assures that the performance of the tuner will be affected solely by changes made to the optimizer. On the other hand, with static data, for any starting point the initial simplex and the consecutive steps are fully deterministic, making reliability test challenging. To address this issue, rather than repeating a number of auto-tuning tests for a given starting point (V_{P_1}, V_{P_2}), we initiate tuning runs for points sampled from a (9×9) pixels region around (V_{P_1}, V_{P_2}) resulting in 81 test runs for each point.

We assess the reliability of the auto-tuning protocol for the 7 experimentally tested configurations listed in Table I (note that for point $(250, 400)$ mV the gate values are adjusted when testing over the pre-measured scan to account for changes in the screening gates). To quantify the performance of the tuner, we define the tuning success rate, P, as a fraction of runs that ended in the "ideal" region (marked with a green triangle in Fig. 4 or in the "sufficiently close" region (marked with a magenta diamond in Fig. 4) with weights 1 and 0.5, respectively. Moreover, in the Network analysis step, we use a neural network with the same architecture as discussed in Sec. II B but trained on a new dataset that includes all three realizations of the SD state. When using optimization parameters resembling those implemented in the lab (i.e., fixed simplex size $\Delta = 75$ mV) and a new neural network, the overall success rate is 45.2 % with standard deviation (*st.dev.*) of 35.5 %. The summary of the performance for each point is presented in Table I (see Table B.II for a comparison of the tuning time and number of iterations between points). Increasing the initial simplex size by 25 mV significantly improves the success rate for all but two points (see column $P_{\Delta=100}$ in Table I), with the overall success rate of 65.2 % (*st.dev.* = 39.4 %). Column $P_{\Delta=f(\delta_0)}$ in Table I shows success rate for tuning when the initial simplex size is scaled based on the fitness value of the initial step, δ_0 such that tuning from points further away from the target area will use a larger simplex than those initiated relatively close to the "ideal" region. The overall success rate here is 74.6 % (*st.dev.* = 31.5 %).

Finally, to assess the performance of the auto-tuning protocol for a wider range of initial configurations, we perform off-line tuning over a set of pre-measured scans. Using four scans spanning 100 mV to 500 mV for plunger P_1 and 150 mV to 550 mV for P_2, measured in 2 mV/pixel resolution, we initiate $N = 784$ test runs per scan, sampling every 10 mV and leaving a margin big enough to assure that the initial simplex is within the full scan boundaries. A heat map representing the performance of the auto-tuner is presented in Fig. 5. As can be seen, the auto-tuner is most likely to fail when initiated with both plunger gates set to either high (above 400 mV) or low (below 300 mV) voltage. While in both cases the "flatness" of the fitness function contributes to the tuning failure, the fixed direction of the initial sim-

plex further contributes to this issue. Adding rotation to the simplex, i.e., varying both plunger gates when determining the second and third step in the optimization (see B and C in Fig. 3), may help with the latter problem.

V. SUMMARY AND OUTLOOK

While a standardized, fully automated approach to tuning quantum dot devices is essential for their scalability, present day tuning approaches rely heavily on human heuristic and algorithmic protocols that are specific to a particular device and cannot be used across devices without fine re-adjustments. To address this issue, we are developing a tuning paradigm that combines synthetic data from a physical model with machine learning and optimization techniques to establish an automated closed-loop system of experimental device control. Here, we reported on the performance of the proposed auto-tuner when tested *in situ*.

In particular, we have verified that, within certain constraints, the proposed approach can automatically tune a QD device to a desired double dot configuration. In the process, we have confirmed that a ML algorithm, trained using exclusively synthetic, noiseless data, can be used to successfully classify images coming from experiment, where noise and imperfections typical for real measurements are present.This work has also enabled us to identify areas where further work is necessary to improve the overall reliability of the auto-tuning system. A new training dataset was necessary to account for all three possible single dot states. The size of the initial simplex also seems to contribute to the mobility of the tuner out of the SD plateau. For comparison, in Table I we present the performance of a tuner using the new network and a bigger simplex size for the experimentally tested starting points. In terms of the length of the tuning runs, at present, the bottleneck of the protocol is the time it takes to perform scans (about 5 min/scan) and the repeated iterations toward the termination of the cycle (i.e., repeated scans of the same region). This can be improved by orders of magnitude by using faster voltage sources and readout techniques and by developing a custom optimization algorithm. Regardless, the power of this new technique lies in its automation, allowing a skilled researcher to spend time elsewhere.

These results serve as a baseline for future investigation of fine-grain device control (i.e., tuning to desired charge configuration) and of "cold start" auto-tuning (i.e., a complete tuning without any pre-calibration of the device). Finally, our work paves the way for similar approaches applied to a wide range of experiments in physics.

To use QD qubits in quantum computers, it is necessary to develop a reliable automated approach to control QD devices, independent of human heuristics and intervention. Working with experimental devices with high-dimensional parameter spaces poses many challenges, from performing reliable measurements to identifying the device state to tuning into a desirable configuration. By combining theoretical, computational, and experimental efforts, this interdisciplinary research sheds new light on how modern ML techniques can assist experiments.

VI. ACKNOWLEDGEMENTS

Research sponsored in part by the Army Research Office (ARO), under Grant Number W911NF-17-1-0274. Development and maintenance of the growth facilities used for fabricating samples was supported by DOE (DE-FG02-03ER46028). We acknowledge the use of facilities supported by NSF through the UW-Madison MRSEC (DMR-1720415). SK gratefully acknowledges support from the JQI-QuICS Lanczos graduate fellowship. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Office (ARO), or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein. Any mention of commercial products is for information only; it does not imply recommendation or endorsement by NIST.

[1] D. Loss and D. P. DiVincenzo. Quantum computation with quantum dots. Phys. Rev. A **57**, 120 (1998).

[2] R. Hanson, L. P. Kouwenhoven, J. R. Petta, S. Tarucha, and L. M. K. Vandersypen. Spins in few-electron quantum dots. Rev. Mod. Phys **79**, 1217 (2007).

[3] F. A. Zwanenburg, A. S. Dzurak, A. Morello, M. Y. Simmons, L. C. L. Hollenberg, G. Klimeck, S. Rogge, S. N. Coppersmith, and M. A. Eriksson. Silicon quantum electronics. Rev. Mod. Phys. **85**, 961 (2013).

[4] 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. Coherent Manipulation of Coupled Electron Spins in Semiconductor Quantum Dots. Science **309**, 2180 (2005).

[5] F. H. L. Koppens, C. Buizert, K. J. Tielrooij, I. T. Vink, K. C. Nowack, T. Meunier, L. P. Kouwenhoven, and L. M. K. Vandersypen. Driven coherent oscillations of a single electron spin in a quantum dot. Nature **442**, 766 (2006).

[6] J. Medford, J. Beil, J. M. Taylor, E. I. Rashba, H. Lu, A. C. Gossard, and C. M. Marcus. Quantum-dot-based resonant exchange qubit. Phys. Rev. Lett. **111**, 050501 (2013).

[7] D. Kim, D. R. Ward, C. B. Simmons, D. E. Savage, M. G. Lagally, M. Friesen, S. N. Coppersmith, and M. A. Eriksson. High-fidelity resonant gating of a silicon-based quantum dot hybrid qubit. npj Quantum Inf. **1**, 15004 (2015).

[8] C Barthel, D. J Reilly, C. M Marcus, M. P Hanson, and A. C Gossard. Rapid single-shot measurement of a singlet-triplet qubit. Phys. Rev. Lett. **103**, 160503 (2009).

[9] 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, and A. S. Dzurak. An addressable quantum dot qubit with fault-tolerant control-fidelity. Nat. Nanotechnol. **9**, 981 (2014).

[10] E. Kawakami, P. Scarlino, D. R. Ward, F. R. Braakman, D. E. Savage, M. G. Lagally, M. Friesen, S. N. Coppersmith, M. A. Eriksson, and L. M. K. Vandersypen. Electrical control of a long-lived spin qubit in a Si/SiGe quantum dot. Nat. Nanotechnol. **9**, 666 (2014).

[11] J. Yoneda, K. Takeda, T. Otsuka, T. Nakajima, M. R. Delbecq, T. Allison, G.Honda, T. Kodera, S. Oda, Y. Hoshi, N. Usami, Kohei M. Itoh, and S. Tarucha. A quantum-dot spin qubit with coherence limited by charge noise and fidelity higher than 99.9 %. Nat. Nanotechnol. **13**, 102 (2018).

[12] M. Veldhorst, C. H. Yang, J. C. C. Hwang, W. Huang, J. P. Dehollain, J. T. Muhonen, S. Simmons, A. Laucht, F. E. Hudson, K. M. Itoh, A. Morello, and A. S. Dzurak. A two-qubit logic gate in silicon. Nature **526**, 410 (2015).

[13] D. M. Zajac, A. J. Sigillito, M. Russ, F. Borjans, J. M. Taylor, G. Burkard, and J. R. Petta. Resonantly driven CNOT gate for electron spins. Science **359**, 439 (2018).

[14] T. F. Watson, S. G. J. Philips, E. Kawakami, D. R. Ward, P. Scarlino, M. Veldhorst, D. E. Savage, M. G. Lagally, Mark Friesen, S. N. Coppersmith, M. A. Eriksson, and L. M. K. Vandersypen. A programmable two-qubit quantum processor in silicon. Nature **555**, 633 (2018).

[15] D. M. Zajac, T. M. Hazard, X. Mi, E. Nielsen, and J. R. Petta. Scalable Gate Architecture for a One-Dimensional Array of Semiconductor Spin Qubits. Phys. Rev. Appl. **6**, 054013 (2016).

[16] U. Mukhopadhyay, J. P. Dehollain, Ch. Reichl, W. Wegscheider, and L. M. K. Vandersypen. A 2×2 quantum dot array with controllable inter-dot tunnel couplings. Appl. Phys. Lett. **112**, 183505 (2018).

[17] T. A. Baart, P. T. Eendebak, C. Reichl, W. Wegscheider, and L. M.K. Vandersypen. Computer-automated tuning of semiconductor double quantum dots into the single-electron regime. Appl. Phys. Lett. **108**, 213104 (2016).

[18] T. Botzem, M. D. Shulman, S. Foletti, S. P. Harvey, O. E. Dial, P. Bethke, P. Cerfontaine, R. P. G. McNeil, D. Mahalu, V. Umansky, A. Ludwig, A. Wieck, D. Schuh, D. Bougeard, A. Yacoby, and H. Bluhm. Tuning methods for semiconductor spin qubits. Phys. Rev. Appl. **10**, 054026 (2018).

[19] C. J. van Diepen, P. T. Eendebak, B. T. Buijtendorp, U. Mukhopadhyay, T. Fujita, C. Reichl, W. Wegscheider, and L. M. K. Vandersypen. Automated tuning of inter-dot tunnel coupling in double quantum dots. Appl. Phys. Lett. **113**, 033101 (2018).

[20] J. D. Teske, S. S. Humpohl, R. Otten, P. Bethke, P. Cerfontaine, J. Dedden, A. Ludwig, A. D. Wieck, and H. Bluhm. A machine learning approach for automated fine-tuning of semiconductor spin qubits. Appl. Phys. Lett. **114**, 133102 (2019).

[21] A. R. Mills, M. M. Feldman, C. Monical, P. J. Lewis, K. W. Larson, A. M. Mounce, and J. R. Petta. Computer-automated tuning procedures for semiconductor quantum dot arrays. Appl. Phys. Lett. **115**, 113501 (2019).

[22] D. T. Lennon, H. Moon, L. C. Camenzind, Liuqi Yu, D. M. Zumbhl, G. A .D. Briggs, M. A. Osborne, E. A. Laird, and N. Ares. Efficiently measuring a quantum device using machine learning. npj Quantum Inf. **5**, 79 (2019).

[23] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural network. L. Bottou F. Pereira, C. J. C. Burges and K. Q. Weinberger (editors), *Advances in Neural Information Processing Systems* (Curran Associates, Inc., New York, 2012), vol. 25, pp. 1097–1105.

[24] S. S. Kalantre, J. P. Zwolak, S. Ragole, X. Wu, N. M. Zimmerman, M. D. Stewart, and J. M. Taylor. Machine learning techniques for state recognition and auto-tuning in quantum dots. npj Quantum Inf. **5**, 6 (2019).

[25] D. M. Zajac, T. M. Hazard, X. Mi, K. Wang, and J. R. Petta. A reconfigurable gate architecture for Si/SiGe quantum dots. Appl. Phys. Lett. **106**, 223507 (2015).

[26] National Institute of Standards and Technology. *Quantum dot data for machine learning.* Data set available at data.gov: `https://catalog.data.gov/dataset/quantum-dot-data-for-machine-learning` (2018).

[27] J. P. Zwolak, S. S. Kalantre, X. Wu, S. Ragole, and J. M. Taylor. QFlow lite dataset: A machine-learning approach to the charge states in quantum dot experiments. PLoS ONE **13**, e0205844 (2018).

[28] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. arXiv:1412.6980 (2014).

[29] J. A. Nelder and R. Mead. A simplex method for function minimization. Comput. J. **7**, 308 (1965).

[30] F. Gao and L. Han. Implementing the Nelder-Mead simplex algorithm with adaptive parameters. Comput. Optim. Appl. **51**, 259 (2012).

[31] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright *et al.* SciPy 1.0–Fundamental Algorithms for Scientific Computing in Python. arXiv:1907.10121 (2019).

[32] QFlow Team. Qflow lite, 2018. Available from: `http://github.com/jpzwolak/QFlow-lite`.

[33] Lab Control Software Scandinavia. Labber, 2019. Version 1.6.3. April 22, 2019. Available from: `http://labber.org`.

Appendix A: Data processing

The model used to simulate the QD devices [27] does not account for noise present in a real measurement. As a result, data used to train the CNN classifier obtained by taking a numerical gradient of the sensor data leads to very clean data, with the background uniformly flattened and charge transition lines clearly visible (see the first column in Fig. A.6). To assure compatibility with the CNN classifier, the acquired experimental scans need to be processed before the probability vector can be assigned to it. Here, the data processing consist of three steps: the nu-

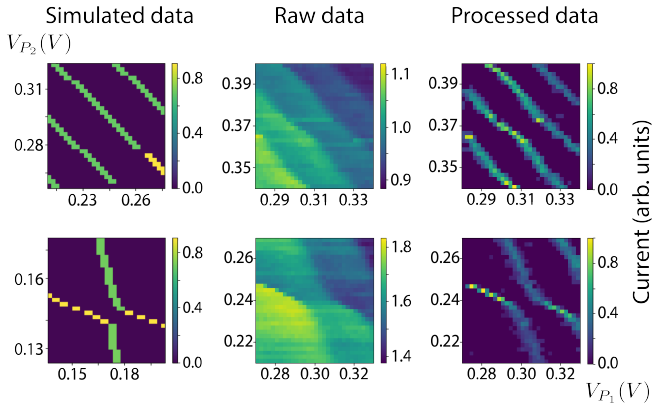

FIG. A.6. Relationship between simulated, raw, and processed data. The top row consists of sample scans with single dot regions and the bottom row of scans with double dot regions. The left column shows the simulated data, the middle columns shows the raw acquired experimental data, and the right column shows the processed experimental data (as "seen" by the CNN classifier).

merical derivative followed by thresholding and re-sizing. To minimize noise, the derivative is taken in the direction of measurement. The gradient data is also tested against unexpected charge sensor flipping and, if necessary, reverted to assure positive values at the charge transition lines. An automated protocol is implemented to normalize the data and to remove the background noise. Finally, the data is re-sized to (30×30) pixels resolution. Second and third column in Fig. A.6 show sample raw and processed data, respectively, for a single and double dot image.

Appendix B: Effect of simplex size on off-line tuning

While varying the simplex size significantly affects the performance of the auto-tuner, leading to an increase in the overall accuracy for the tested points by nearly 40 % (see Table I for details), it did not affect the number of iterations of the optimizer. In particular, the overall average number of iterations for the three tested simplex sizes was: 13.3 (pooled $st.dev. = 2.5$), 13.7 (pooled $st.dev. = 2.3$), and 13.6 (pooled $st.dev. = 2.3$) for tuning with initial size of $\Delta = 75\,\mathrm{mV}$, $\Delta = 100\,\mathrm{mV}$, and $\Delta = f(\delta_0)$, respectively. Table B.II shows the average number of iterations executed by the optimizer for each tested point.

TABLE B.II. Average (standard deviation in parentheses) number of iteration when tuning off-line for varying configuration of the initial simplex Δ. In all cases, the average is taken over $N = 81$ test runs for points sampled within 10 mV around each experimentally tested point given by (V_{P1}, V_{P2}).

	$\Delta = 75\,\mathrm{mV}$	$\Delta = 100\,\mathrm{mV}$	$\Delta = f(\delta_0)$
(250,400)	12.7 (2.5)	12.2 (2.0)	12.6 (2.2)
(350,400)	14.0 (2.4)	13.6 (2.2)	13.5 (2.3)
(350,415)	13.2 (2.3)	14.1 (2.1)	13.4 (2.1)
(350,425)	12.9 (2.3)	13.9 (2.1)	13.6 (2.2)
(350,450)	11.6 (2.7)	13.3 (2.4)	13.9 (2.5)
(400,350)	13.9 (2.3)	14.0 (2.2)	13.3 (1.8)
(450,350)	14.5 (2.6)	15.0 (2.6)	15.0 (2.5)

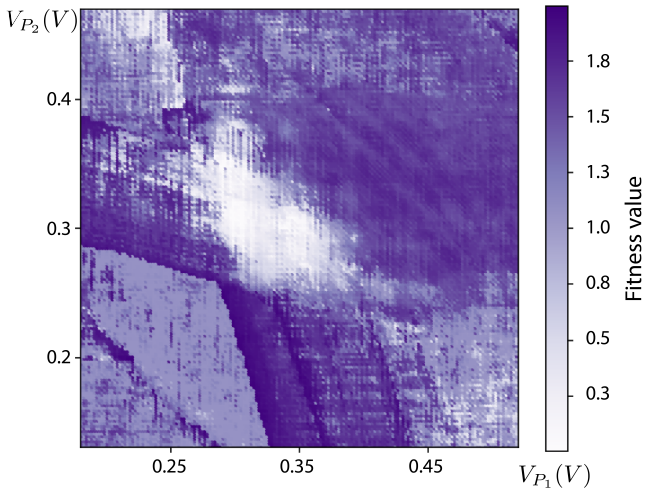

FIG. C.7. Fitness function over a sample device shown in Fig. 4.

Appendix C: Fitness function

We plot the fitness value for tuning to a double dot regime as a function of plunger gate voltages for a scan with experimental data. In particular, for each point in the voltage space, as presented in Fig. 4, we calculate the fitness value for a region centered at this point. This allows to represents the landscape over which the auto-tuning optimization runs (a 171 pixels map). One can see the double dot state forming a minimum near the center of Fig. C.7 which represents the target area for tuning.