Hamiltonian learning using machine-learning models trained with continuous measurements

Kris Tucker,[1,*] Amit Kiran Rege,[2,†] Conor Smith,[3,4] Claire Monteleoni,[2,5] and Tameem Albash[6]

[1] *Department of Applied Mathematics, University of Colorado Boulder, Boulder, Colorado, USA*
[2] *Department of Computer Science, University of Colorado Boulder, Boulder, Colorado, USA*
[3] *Center for Quantum Information and Control, University of New Mexico, Albuquerque, New Mexico, USA*
[4] *Department of Electrical and Computer Engineering, University of New Mexico, Albuquerque, New Mexico, USA*
[5] *INRIA Centre de Recherche de Paris, Paris, France*
[6] *Center for Computing Research, Sandia National Laboratories, Albuquerque, New Mexico 87185, USA*

(Received 27 April 2024; revised 19 August 2024; accepted 26 September 2024; published 30 October 2024)

We build upon recent work on the use of machine-learning models to estimate Hamiltonian parameters using continuous weak measurement of qubits as input. We consider two settings for the training of our model: (1) supervised learning, where the weak-measurement training record can be labeled with known Hamiltonian parameters, and (2) unsupervised learning, where no labels are available. The first has the advantage of not requiring an explicit representation of the quantum state, thus potentially scaling very favorably to a larger number of qubits. The second requires the implementation of a physical model to map the Hamiltonian parameters to a measurement record, which we implement using an integrator of the physical model with a recurrent neural network to provide a model-free correction at every time step to account for small effects not captured by the physical model. We test our construction on a system of two qubits and demonstrate accurate prediction of multiple physical parameters in both the supervised context and the unsupervised context. We demonstrate that the model benefits from larger training sets, establishing that it is "learning," and we show robustness regarding errors in the assumed physical model by achieving accurate parameter estimation in the presence of unanticipated single-particle relaxation.

DOI: 10.1103/PhysRevApplied.22.044080

I. INTRODUCTION

As the scale, complexity, and availability of quantum devices continues to grow over the coming decades [1], the ability to accurately characterize device parameters, especially unwanted effects, will become ever-more important. For example, nonlocal errors such as crosstalk in large systems [2,3] can be problematic for quantum error correction methods [4–8]. Detection of these unwanted effects is a critical first step in mitigating them, with efforts to do so complicated by the exponential scaling of the dimension of the Hilbert space with system size and the challenges of modeling increasingly complex systems.

Recent advances in the application of machine-learning (ML) tools to quantum systems have shown promise in overcoming some of these difficulties. Scalable ML models have been used to represent quantum states for state tomography [9–14] and evolution [14,15], for learning unknown dynamics [16–18], and for learning device characterization from measurements [19–23]. Increasingly, *a priori* knowledge of a physical system has been combined with ML models to increase accuracy and interpretability [24]. The model-free nature of many ML solutions has made them scalable and robust regarding many of the common pitfalls of physical models, such as non-Markovian dynamics [15,19–21]. There are trade-offs, however, as more abstract models are generally less interpretable, which limits the physical insights that can be gleaned from them. Finding a balance between these two competing properties, representability and interpretability, is thus an important challenge.

In this work, we build upon recent advances in device characterization applying ML models to the continuous measurement of qubits [19–21]. In contrast with the parameter-estimation approach in Ref. [19], where a stochastic master equation (SME) is used as a trainable model, our ML model learns a direct map from continuous measurement inputs to the system parameters of interest. In the case where the measurement records are associated with known device parameters (the case of supervised learning), this approach has two advantages. First, the

*Contact author: kristopher.tucker@colorado.edu, ktucker27@gmail.com
†Contact author: amit.rege@colorado.edu

model is completely independent of any state representation (in contrast with approaches such as the approach in Ref. [22]). The measurements need only provide enough information to estimate the parameters of interest and not the complete quantum state, potentially freeing the model from the curse of dimensionality that comes from the exponential scaling of the dimension of the Hilbert space and thus making it more scalable.

Second, once the ML model is trained, parameter estimation can be performed quickly even for systems with parameters not seen during training. It can also benefit from having been provided with many examples of noisy measurement records associated with known true values in the training set. This makes it possible for the model to learn to distinguish between those features of measurement records that vary with parameters and those that are uncorrelated noise, thus potentially requiring fewer measurements for estimation.

In the event that device parameters are not provided along with measurement records, an unsupervised approach comparing the input with a measurement record reconstructed from parameter estimates can be used. This requires a map from parameter estimates to measurement output, which is accomplished by addition of a layer combining an integrator of the physical model with a recurrent neural network (RNN) to provide a model-free correction at every time step to account for unanticipated effects, as long as they are consistent and small relative to the dynamics driven by the parameters of interest. This uses the capabilities of neural ordinary differential equations (ODEs) found in other studies [15,25,26] but enhanced to provide a completely-model-free correction not bound by assumptions of linearity or Markovianity, with a projection step to return the corrected state to the manifold of physical density operators [27]. The output of the model is then the estimated solution to the unconditioned master equation with learned corrections for effects beyond the master equation, which can be compared with the measurement records to update model parameters. While this case is not completely model-free, it also does not make rigid assumptions about the dynamics and remains capable of accurate parameter estimation in the presence of unanticipated effects that would otherwise severely impact accuracy. The ML model is not burdened with learning all of the physics; it just has to correct for small effects the model may have missed, an approach known as "discrepancy modeling" [28,29].

The fully unsupervised model can be viewed as a denoising autoencoder [30] where the physical parameters being estimated take on the role of the latent space, with interpretability enforced by the presence of the physical model in the decoder. The encoder is the map from measurement records to parameters, and is the desired product of the training to be used for fast and accurate parameter estimation.

While the unsupervised model clearly has applications for parameter estimation in cases where nothing is known about the parameters of interest, the supervised approach using just the encoder could still find applications for systems where parameters are known at the time training data are generated, but a prediction routine is still required in other circumstances, as would be the case for detecting drift away from device calibration over time. We apply both approaches to the specific problem of learning one- and two-body Hamiltonian parameters in a two-qubit system.

Finally, we contrast our work with recent methods for Hamiltonian learning based on Gibbs state measurements or real-time evolution [22,31–38]. In our work the model is trained and performs predictions based on continuous measurements modeled by a stochastic master equation rather than Hamiltonian or deterministic-master-equation evolution combined with strong measurements. Our approach, which is applicable to both Hamiltonian and Lindblad parameters, requires that the parameters to be learned have an observable impact on the continuous measurements, and it does not use the preparation of a steady state or energy eigenstate [22,32,38] or changing the dynamics by adaptively changing the Hamiltonian [36] or introducing additional dynamics [34,35]. It is likely adaptive approaches could enhance our learning as well. Finally, the discrepancy modeling in our approach is unique in that it allows the combination of knowledge of the physical system with model-free machine learning to increase estimation accuracy and ML model interpretability.

This article is organized as follows. In Sec. II, we describe the physical system of interest to us, which is given by two qubits that are weakly measured. In Sec. III, we describe our ML models and their supervised and unsupervised training. In Sec. IV, we show results for the performance of the models. In Sec. V, we summarize our results and discuss possible future extensions of this work.

II. PHYSICAL SYSTEM

Our physical system of interest is that of two qubits with fixed position in a microwave cavity as illustrated in Fig. 1.

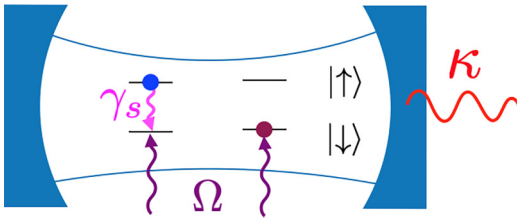

FIG. 1. The physical system of two qubits in a microwave cavity subjected to a Rabi drive Ω and subject to single-particle relaxation with rate γ_s and dephasing due to weak-measurement back action with rate κ.

A single common mode of the cavity is coupled to the computational degree of freedom of the qubits, and they are coherently driven on resonance with a Rabi drive of frequency Ω. A two-qubit interaction term is present with magnitude ϵ. A weak measurement tone [39,40] is applied to the cavity to probe the qubit state in one of the $\{X, Y, Z\}$ directions for each qubit, with measurement-back-action dephasing rate κ. On adiabatic elimination of the cavity mode [41], which we assume has dynamics evolving at a rate much faster than timescales relevant to the qubits, we can describe the system with the SME [42,43]

$$d\rho = -i[H,\rho]dt + \sum_{i=1}^{2}\mathcal{D}[L_i](\rho)dt + \sum_{i=1}^{2}\sqrt{\frac{\eta}{2}}\mathcal{H}[L_i](\rho)dW_t^{(i)}, \tag{1a}$$

$$H = \sum_{i=1}^{2}\frac{\Omega}{2}X_i + \epsilon Z_1 Z_2, \tag{1b}$$

where $\{X_i, Y_i, Z_i\}$ is the set of Pauli operators for qubit $i \in \{1,2\}$, $\mathcal{D}[L](\rho) = L\rho L^\dagger - \frac{1}{2}\left\{L^\dagger L\rho + \rho L^\dagger L\right\}$ is the Lindblad superoperator, $\mathcal{H}[L](\rho) = L\rho + \rho L^\dagger - \rho\mathrm{Tr}[\rho(L + L^\dagger)]$ is the measurement superoperator, η is the efficiency of the measurement, and $L_i = \sqrt{\kappa}C_i$, where $C_i \in \{X_i, Y_i, Z_i\}$ is the weak-measurement operator. Here we have suppressed the dependence on t for ρ, and we have adopted the normalization convention $\hbar = 1$.

The stochastic differential equations (SDEs) for the measurement records are given by

$$dr_i = \sqrt{\frac{\eta}{2}}\mathrm{Tr}\left[\rho\left(L_i + L_i^\dagger\right)\right]dt + dW_t^{(i)}, \tag{2}$$

where r_i is the weak-measurement record for qubit $i \in \{1,2\}$ and the independent Wiener increments $dW_t^{(i)}$ are the same as those appearing in Eq. (1a). This is comparable to the system studied in Ref. [19], except generalized to two qubits and with the addition of the two-qubit interaction term in Eq. (1b). We also consider cases where the weak-measurement operator L_i is different for each qubit.

The unconditioned master equation is obtained by one averaging over all possible trajectories of the Wiener processes and is given by

$$d\overline{\rho} = -i[H,\overline{\rho}]dt + \sum_{i=1}^{2}\mathcal{D}[L_i](\overline{\rho})dt, \tag{3a}$$

$$d\overline{r_i} = \sqrt{\frac{\eta}{2}}\mathrm{Tr}\left[\overline{\rho}\left(L_i + L_i^\dagger\right)\right]dt. \tag{3b}$$

III. MACHINE-LEARNING MODEL

A. Preliminaries

We briefly review the kinds of architecture used in this work. For a more-detailed overview, see Appendix A.

RNNs [44] are a class of artificial neural networks designed to recognize patterns in sequences of data. Unlike traditional neural networks, which assume all inputs (and outputs) are independent of each other, RNNs are characterized by their ability to retain information from previous inputs in the sequence through internal memory, making them ideal for tasks involving sequential data.

Traditional RNNs struggle to learn and retain information over long sequences due to problems such as the contribution of information decaying over time [45]. This makes it hard for the network to maintain long-term dependencies. Long-short-term-memory (LSTM) networks [46] are a type of RNN architecture designed to address the issue of long-term dependencies in sequence data. In this work, we use LSTM-based models to model trajectories of weak measurements.

We now describe a common design recipe used in tasks performing unsupervised translation of data from one domain to another, often referred to, collectively, as the "family of encoder-decoder models." These are also called "sequence-to-sequence models" when working with sequential data. Such models produce an arbitrary length, context-dependent sequence, given an input sequence. The length of the input sequence need not match the length of the output sequence, and individual elements of the input sequence need not have one-to-one correspondence with individual elements of the output sequence.

The key idea is to use the encoder network to obtain a compact representation of the input sequence (often called the "context vector"). This representation is then passed to the decoder network to produce an output sequence by adding it to the hidden state in the decoder. This is an extremely general framework, and potentially any neural network can be used in such an arrangement (see Fig. 2).

In our unsupervised setup, the encoder is an LSTM that returns the master-equation parameters as the context. The decoder is a numerical ODE integrator implemented as an RNN.

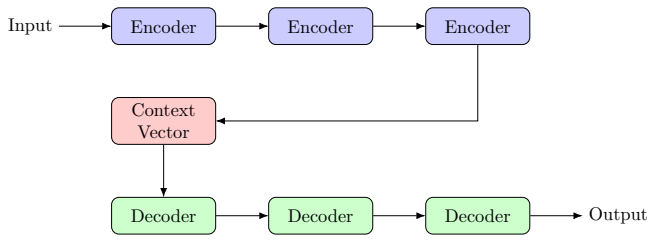

FIG. 2. A high-level description of the encoder-decoder design recipe.

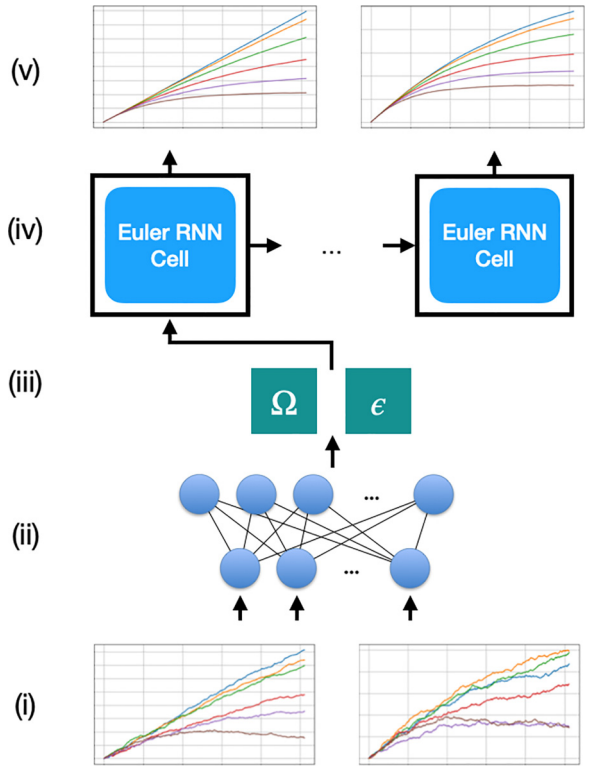

FIG. 3. Full ML model. Voltage records are measured from the cavity system producing noisy averaged voltage records (i), and these are sent through a neural network encoder (ii) producing system parameters as output (iii). These parameters are used by the flex integrator decoder (iv) to produce noise-free voltage estimates (v).

B. Model description

Our objective is to train a ML model to estimate the physical master-equation parameters in Eqs. (1a) and (1b) by observing the weak-measurement records described by Eq. (2). Input data are provided for training of the ML model in the form of a collection of *averaged* weak-measurement records, where the average is over some number of records d to reduce the effects of diffusive noise. The output of the model is either (1) the physical master-equation parameters of interest in the supervised case, where labeled training data are available, or (2) the predicted measurement record satisfying the unconditioned master equation [Eqs. (3a) and (3b)] in the unsupervised case, where true parameters are not available. The former configuration can be interpreted as an encoder mapping noisy trajectories to estimated parameter values, while the latter configuration adds a decoder that takes the estimated parameters as input and produces a predicted solution to the unconditioned master equation. The full architecture is shown in Fig. 3 and can be viewed as a denoising autoencoder taking noisy input trajectories and producing output with the noise removed.

The encoder begins with an average pooling operation in the time dimension to further smooth the trend component of the voltages, followed by an LSTM layer [47] to process the sequential input data and consolidate information from multiple qubits into a single sequence. This is followed by a feed-forward neural network of dense layers ending with the parameter layer. The LSTM and feed-forward layers constitute the neural network shown in Fig. 3(ii).

In the unsupervised case, interpretability of the latent variables as parameters is enforced by use of them directly in an enhanced numerical ODE integrator in the decoder for the unconditioned master equation. The integrator is implemented as an RNN with a custom cell that combines a single step of Euler's method with a correction produced by a standard LSTM cell. This is shown in Fig. 4. The correction term is designed to compensate for unanticipated dynamics in the physical model used by the numerical integrator. The design leverages the concept of a neural ODE [25] or an ODE-RNN [26,47], as it uses a neural network to approximate corrections to the unconditioned master equation as

$$d\overline{\rho} = f_{\text{model}}(t, \overline{\rho}; \theta)\, dt + f_{\text{LSTM}}(t, \overline{\rho}; \theta)\, dt, \tag{4}$$

where θ is a set of ODE parameters, f_{model} is the drift term of the form of Eq. (3), and f_{LSTM} is an LSTM cell and as such is a nonlinear, model-free function of the time, state, and parameters.

The carry state used in the standard LSTM architecture is passed between evaluations of f_{LSTM} at each time point, which means information from the full history of states is potentially available, allowing the modeling of non-Markovian effects. We ensure that the output operator is a quantum state at each time step by restricting the free parameters in the correction to have a Hermitian form to enforce Hermiticity, by implementing a normalization step to ensure that the trace is preserved, and by performing an orthogonal projection of the operator back onto the

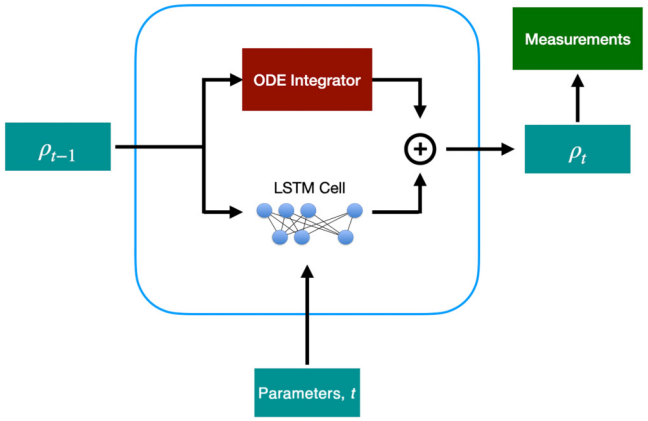

FIG. 4. Details of the decoder RNN cell combining a standard ODE integrator with an LSTM update.

state space according to the algorithm presented in Ref. [27]. In this respect, it is analogous to the approach used by the time-dependent-variational-principle algorithm [48,49] for modeling dynamics of matrix product states (MPSs) [50,51].

We emphasize that the enhanced numerical integrator primarily relies on a known physical model given by f_{model} and uses the ML components in f_{LSTM} only to compensate for discrepancies between this model and the experimental system, which are assumed to be small relative to the known dynamics. As such, it could be categorized along with other approaches in ML for discrepancy modeling [28,29] that operate under the principle that small corrections are easier to learn than the full dynamics. A similar approach is taken to modeling non-Markovian dynamics via RNNs in Ref. [15], although our approach differs in that it combines discrepancy modeling with a completely general, nonlinear correction to the state evolution based on a neural ODE. In addition, as suggested in the outlook section of Ref. [15], our investigation focuses on the situation where only a subset of the measurement information necessary for full state reconstruction is available, so training-data-volume requirements are substantially reduced.

Although the integrator used in the decoder for this work is an Euler integrator for the unconditioned master equation, this piece is completely modular in design and can be replaced with any integrator for simulating quantum dynamics. For example, a time-dependent-variational-principle integrator using MPSs could be used for larger systems that are approximated well by an MPS ansatz, with the trainable decoder parameters accounting for errors introduced by this approach. It should also be noted that, since the output of the decoder is the solution to the unconditioned master equation, it contains an ODE integrator rather than an SDE integrator. Once it has been trained, however, it would be possible to switch to an SDE integrator, for example, by one switching from an Euler integrator to an Euler-Maruyama integrator, by simulating noise. The model would then serve as a generative model that could be useful for simulation purposes.

C. Model interpretation

The output of the decoder is the estimated solution to the unconditioned master equation [Eq. (3b)], and the loss is calculated as the mean squared error (MSE) between this estimate and provided approximations of clean measurements used as label values in the case where true parameter estimates are not available. Thus, it is desirable that these labels contain as little noise as possible, although estimation can still occur with noisy labels, as seen in Sec. IV. Clean label values are realized by one averaging over many input groups sharing the same parameter values and using this same average for all of the contributing input trajectory groups during training. See Sec. III D for more details. In this way, the model can be viewed as a denoising autoencoder [30], with denoised output provided for every noisy input example, and with a latent space corresponding to the physical parameters being estimated.

In our model, the parameters learned during training are the various weights and biases of the neural networks in the encoder and the decoder. Notably, we are not directly estimating master-equation parameters during training, but rather learning a function that maps noisy measurement records to physical parameters. This approach differs from that in Ref. [19], where the physical parameters are the result of an optimization and no such function is learned. One advantage of our approach is that it enables extremely fast parameter estimation once the model is trained, even when observing systems with parameter sets not included in the training data. Furthermore, in the case where the training data include true physical-parameter values, it allows us to learn the map with the encoder only, bypassing the need for a physical model entirely. In this case, scaling is limited by the amount and type of training data required, rather than system size. The amount of training data required will vary by application, and requires further investigation. Finally, learning a map for physical parameters as a function of measurement records allows the possibility of fewer measurements being required for prediction versus training as the model learns to account for noise in the input data, as illustrated by results in Sec. IV showing how estimation accuracy increases with training-set size even when the number of input records is held constant.

A second difference between our approach and that in Ref. [19] is that when labeled training data are unavailable and a physical decoder is necessary, the flexible correction scheme in the decoder allows the model to compensate for non-Markovian or nonlinear dynamics since it is not bound by a Lindblad form, while still being informed by the master equation for a first-order model of the dynamics.

D. Training and data

To generate the data for training, validation, and testing, we use an Euler-Maruyama integrator to solve Eq. (2) for both qubits for each of 40 values of ϵ evenly spaced on $[0, 2)$ with fixed $\Omega = 1.395$ rad/μs, and the same number of Ω values evenly spaced on $[1, 5)$ with fixed $\epsilon = 1.0$ rad/μs for a total of $K = 80$ (Ω, ϵ) pairs. For each pair of values (Ω_k, ϵ_k), $N = 32\,000$ measurement trajectories are simulated for $T = 4$ μs. Half of the collection of pairs of values (Ω_k, ϵ_k), corresponding to every other parameter pair, is used for training. The other half, corresponding to parameters midway between training values, is split evenly in the number of trajectories to be used for validation and testing. This ensures the training set

contains a disjoint set of parameter pairs from the validation and test set. The two end points of the validation and test set are excluded from each end to ensure that the training data extend slightly beyond the domain of validation and test values. Only weak-measurement records are being used, and no strong measurements are being performed.

Values of $\kappa = 3.326$ rad/μs and $\eta = 0.1469$ are selected to be consistent with the example of superconducting qubits as found, for example, in Ref. [19], although with a stronger measurement back action κ after simulations showed a larger value offers a good balance between dephasing rate and measurement noise. While either Ω or ϵ or both are treated as unknown parameters and constitute the latent space of our model, it is assumed in our simulations that κ and η are known with high precision as they have already been calibrated. Our approach can be used to perform this calibration step, since the latent space of the model is not limited to Hamiltonian parameters and could support any combination of Hamiltonian and Lindblad parameters as unknown variables. Alternatively, other calibration methods found in the literature can be used to estimate them, such as in Ref. [52]. It should be noted, however, that in practice if calibration errors are present in the model values, the degrees of freedom in the decoder should make parameter estimation robust regarding these errors as demonstrated in what is likely the more-challenging case of unanticipated single-particle relaxation examined in Sec. IV C. The impact on parameter-estimation accuracy is expected to be small for small calibration errors. A study quantifying this dependency is an interesting area for future work.

During training, for each true parameter set $\theta_k = (\Omega_k, \epsilon_k)$, trajectory groups of a preset size d are randomly selected from the full training set and their averages are provided as input values to the model, such that each minibatch comprises $M = N/d$ averaged trajectories $\{\tilde{x}_{j,k}(t)\}_{j=1}^{M}$ where $\tilde{x}_{j,k}(t) = (1/d)\sum_{i\in I} r_{i,k}(t)$, $r_{i,k}(t)$ is measurement record i for parameter set k, and I is a set of trajectory indices of size d randomly selected from $1, 2, \ldots, N$ without replacement until all N trajectories are used, which defines one epoch of training. In this way, each minibatch consists of a different set of noisy trajectories as input to maximize the diversity of training examples. In the unsupervised case, for all $j = 1, \ldots, M$, the value $x_k(t)$ to be used in the loss function is the average over the full training set of trajectories associated with the true parameter set θ_k, such that $x_k(t) = (1/N)\sum_{i=1}^{N} r_{i,k}(t)$.

To make the interpretation of our model as a denoising autoencoder more concrete, we note that denoising autoencoders are characterized by a corruption process $C(\tilde{y}|y)$ whereby noisy inputs $\tilde{y}$ are generated for each uncorrupted training example y [30]. In our case, the full trajectory means $x_k(t)$ takes on the role of y, approximating the solution to the unconditioned SME, while the random selection of much-smaller trajectory groups is the process by which the corrupted data elements $\tilde{y}$ are generated.

In the supervised case, the loss function is the MSE between predicted parameter sets $\tilde{\theta}_k$ and the provided true values θ_k:

$$\mathcal{L} = \frac{1}{pK}\sum_{k}\left|\left|\tilde{\theta}_k - \theta_k\right|\right|_2^2, \tag{5}$$

where p is the dimension of θ_k, the number of parameters being estimated. This is further averaged over the M groups of size d when multiple trajectory groups are provided as input. In the unsupervised case, the loss function is the MSE between each x_k and the estimated clean measurement record:

$$\mathcal{L} = \frac{1}{MKN_t}\sum_{j,k,t}\left|x_k(t) - \mathcal{M}(\tilde{x}_{j,k})(t)\right|^2, \tag{6}$$

where $\mathcal{M}$ denotes the model, t is the time index, and $N_t \equiv T/\Delta t$ is the number of time points excluding the initial condition.

During evaluation, validation and test error is evaluated in a manner similar to how training minibatches are selected, with groups of size d randomly chosen from the full validation and test sets, followed by the calculation of the MSE. This process is repeated 100 times and the average is taken as the calculated MSE for each set. Note that for some scenarios only one group will be available as input, either because the data is noise free or because the group size equals the dataset size, in which case the shuffling has no effect.

Training is performed for multiple runs of 100 epochs, with a learning rate of 3×10^{-3} and a decay rate of 0.99 per epoch, with the learning rate reset after each run, continuing until the validation loss drops by less than 5% from run to run, and for the last 20 epochs of the final run. Once converged, the MSE is computed for the validation and test sets. This entire process is performed 100 times, each with a different random initialization of the model parameters. The best model is considered to be the one with the smallest validation loss after the final epoch, and this model is used to evaluate the MSE for the test set. Hyperparameter tuning for the model layer sizes is performed with a grid search for a physical system with parameters distinct from those in Sec. IV.

IV. EVALUATION

To fully assess the performance of the model, we evaluate two distinct cases: the encoder alone to evaluate the case of labeled training data, and the full model for the case where labels are not available. We consider a range of trajectory group sizes d as well as noise-free data derived

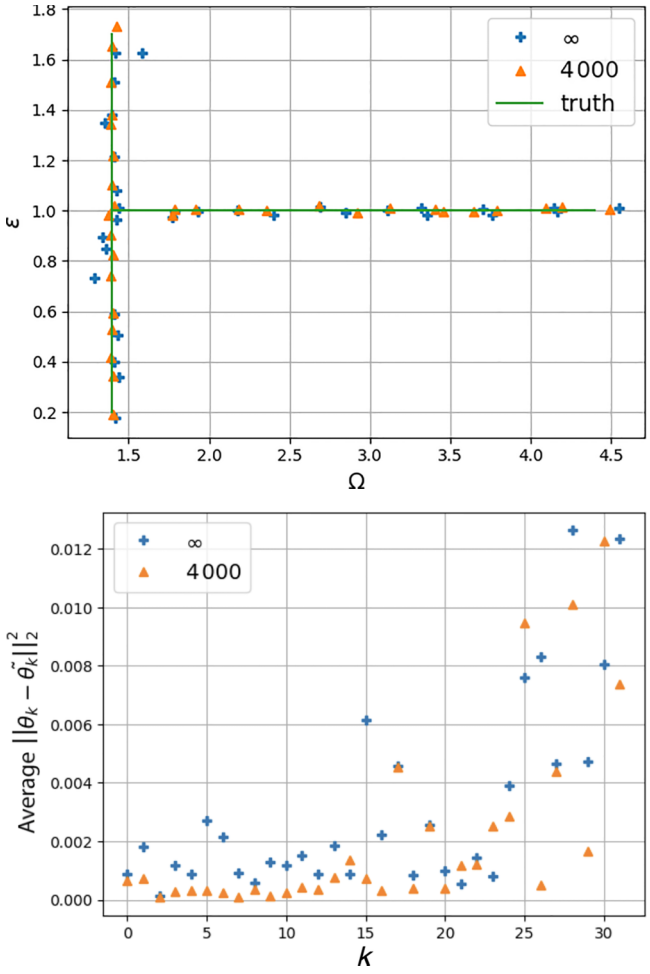

FIG. 5. Estimations of the parameter pair $\theta_k = (\Omega_k, \epsilon_k)$ (top) and the squared error (bottom) averaged over four noisy trajectory groups of $d = 4000$ measurement records for a model trained on clean data and noisy data, then evaluated on noisy data. Shuffle-evaluated test-set MSEs are $(8.67 \times 10^{-3}, 5.83 \times 10^{-4})$ and $(4.94 \times 10^{-3}, 4.87 \times 10^{-4})$, respectively.

from an Euler integration of the unconditioned master equation. The noise-free case is denoted by ∞ as the number of trajectories in all tables.

To minimize the impact of the weak-measurement back action via the parameter κ, in Secs. IV A and IV B the initial state is chosen to be spin-up in the directions of measurement, and the X and Y directions are used for the first qubit and the second qubit, respectively. A different configuration is used in Sec. IV C, where more-diverse measurements are needed to correct for unanticipated single-particle relaxation not explicitly present in the decoder's physical model.

A. Supervised learning

First, we study the impact of noise in the training set and how it helps or hinders the model's performance when predicting parameters from noisy data. We do this by performing estimation with two models, one trained with noise-free measurement values, and the other set trained with eight groups of $d = 4000$ averaged measurement records. Both models are then evaluated on a test set of noisy, averaged measurement records with the same group size d as the noisy training set. In both cases, the model with the best validation loss on its respective validation set out of 100 randomly initialized models is used.

Figure 5 shows an example of the error of the estimated parameter pair $\tilde{\theta}_k$ compared with truth for each of the 32 k values in the trimmed test set. We see that for many specific pairs and for the overall MSE, the model trained on the noise-free measurements produces less-accurate estimates than the model trained on noisy data. This is consistent in the Ω and ϵ errors. This suggests that the model learns to account for noise, as expected, and training with noise on the level of measurements used for prediction is beneficial.

Next we examine the impact of the training group size d. Table I lists test-set MSEs for a number of training sets where evaluation is done with the same group size d used for training. The best and median were taken from the list of results sorted according to validation loss, as this is what would be known in practice during training. Here we see a steady improvement as the number of trajectories in a group increases, and noise decreases, saturating in the case where noise-free data are used for training and evaluation. We note that this test differs from the test corresponding to the results in Fig. 5, where the model trained without noise was evaluated on noisy data, as here we consider only the case where training and test group sizes are equal.

TABLE I. MSE pair for (Ω, ϵ) estimates on the supervised training set with only the encoder used.

d	Best MSE	Median MSE	Mean MSE
2000	$9.87 \times 10^{-3}, 8.76 \times 10^{-4}$	$9.72 \times 10^{-3}, 8.71 \times 10^{-4}$	$9.75 \times 10^{-3}, 9.79 \times 10^{-4}$
4000	$4.94 \times 10^{-3}, 4.87 \times 10^{-4}$	$5.23 \times 10^{-3}, 5.44 \times 10^{-4}$	$5.62 \times 10^{-3}, 5.75 \times 10^{-4}$
8000	$4.17 \times 10^{-3}, 2.74 \times 10^{-4}$	$2.94 \times 10^{-3}, 2.74 \times 10^{-4}$	$3.38 \times 10^{-3}, 4.13 \times 10^{-4}$
16 000	$2.19 \times 10^{-3}, 2.06 \times 10^{-4}$	$1.40 \times 10^{-3}, 1.92 \times 10^{-4}$	$1.80 \times 10^{-3}, 2.94 \times 10^{-4}$
∞	$1.01 \times 10^{-5}, 2.04 \times 10^{-5}$	$4.51 \times 10^{-6}, 4.82 \times 10^{-5}$	$1.32 \times 10^{-5}, 5.68 \times 10^{-5}$

TABLE II. MSE of (Ω, ϵ) when $d = 4000$ trajectories are used to estimate parameters for differing training-set sizes N for an unsupervised training set with fixed Ω and varying ϵ. The values of κ and η are assumed to be known exactly.

N	Best MSE	Median MSE	Mean MSE
4000	1.51×10^{-3}, 1.12×10^{-3}	6.58×10^{-4}, 9.86×10^{-4}	8.33×10^{-4}, 1.08×10^{-3}
8000	3.27×10^{-4}, 7.71×10^{-4}	5.20×10^{-4}, 7.94×10^{-4}	5.38×10^{-4}, 8.81×10^{-4}
16 000	1.83×10^{-4}, 7.49×10^{-4}	3.20×10^{-4}, 8.41×10^{-4}	3.07×10^{-4}, 8.38×10^{-4}
32 000	8.14×10^{-5}, 7.16×10^{-4}	9.82×10^{-5}, 8.15×10^{-4}	1.45×10^{-4}, 8.12×10^{-4}

B. Unsupervised learning

We now consider the unsupervised case, where the labels of the training data are not known, but it is assumed that the physical model in the decoder is correct, i.e., we are still not learning parameters for the drift function correction f_{LSTM}, which we consider in Sec. IV C. We are therefore using the MSE in Eq. (6) as the loss function.

First we examine how parameter-estimation accuracy for a fixed group size $d = 4000$ varies with the total training-set size N. Table II shows (Ω, ϵ) test-set MSEs for the case where Ω is unknown but fixed at a true value of 1.395 rad/μs and ϵ is allowed to vary, for various training set sizes N. The same test set containing 16 000 trajectories is used for each row. From the Table II, we see that accuracy increases significantly as the amount of training data increases, even though the groups being presented to the model for parameter estimation remain the same. This indicates that a greater diversity of noisy measurement records when training results in models that can produce more-accurate parameter estimates when presented with the same number of measurements when performing prediction. This motivates the use of $N = 32\,000$ going forward.

Next we consider the impact of group size d and measurement-record time spacing Δt on accuracy. Table III shows (Ω, ϵ) test-set MSEs for the case where Ω is unknown but fixed at a true value of 1.395 rad/μs and ϵ is allowed to vary. It illustrates how extremely low test-set MSEs are achievable for Ω in this case, which is expected given the high volume of training data available for a single value. The accuracy of ϵ estimates depends on both the number of trajectories used to create each input sequence and the time spacing at which measurements are recorded. To evaluate measurement records with differing time spacing, trajectories simulated with $\Delta t = 2^{-8}$ μs are subsampled to avoid the introduction of numerical integration error associated with simulations with a larger time step.

Table IV shows test-set MSEs for the training set where both Ω and ϵ are allowed to vary. This is a harder task as the model has fewer training data for each unique Ω value, hence the loss in accuracy for that parameter. The error in ϵ remains roughly the same as or better than in the fixed-Ω case, with a best-case root-mean-square error of around 1% of the median test value of ϵ. Here we see a roughly linear trend in the MSE versus d at smaller time steps, as doubling of the number of trajectories in a group roughly halves the MSE. This trend breaks down, however, for the $\Delta t = 2^{-4}$ μs case, suggesting a permissive time-step threshold around $\Delta t = 2^{-6}$ μs at or below which the expected trend in accuracy versus input-data size is realized.

C. Model correction

In this section we demonstrate the ability of the decoder to correct for dynamics not explicitly considered in the physical model [Eq. (1a)]. This is done by our enabling training for the decoder LSTM parameters. For the datasets, we simulate 30 000 trajectories with fixed $\Omega = 1.395$ rad/μs, which is known to the model, but with varying unknown ϵ, which is the parameter to be estimated. We add a dissipative term for each qubit corresponding to the Lindblad operator $\mathcal{D}[\sqrt{\gamma_s}\sigma_i^-](\rho)$ in Eq. (1a), where $\gamma_s = 0.1$ and $\sigma_i^- = \frac{1}{2}(X_i - iY_i)$ is the single-particle relaxation operator mapping the excited state to

TABLE III. MSE of (Ω, ϵ) estimates on the unsupervised training set with fixed Ω and varying ϵ and $N = 32,000$. The values of κ and η are assumed to be known exactly.

d	$\Delta t = 2^{-8}$ μs	$\Delta t = 2^{-6}$ μs	$\Delta t = 2^{-4}$ μs
2000	5.78×10^{-5}, 1.40×10^{-3}	1.20×10^{-4}, 1.40×10^{-3}	1.22×10^{-4}, 2.83×10^{-3}
4000	8.14×10^{-5}, 7.16×10^{-4}	1.66×10^{-4}, 7.90×10^{-4}	1.19×10^{-4}, 2.09×10^{-3}
8000	8.51×10^{-5}, 3.42×10^{-4}	7.16×10^{-5}, 3.63×10^{-4}	8.02×10^{-5}, 1.82×10^{-3}
16 000	8.31×10^{-5}, 1.57×10^{-4}	1.66×10^{-4}, 1.57×10^{-4}	1.19×10^{-4}, 1.89×10^{-3}
∞	1.66×10^{-6}, 6.08×10^{-6}	7.21×10^{-6}, 1.04×10^{-4}	1.52×10^{-5}, 1.71×10^{-3}

TABLE IV. MSE of (Ω, ϵ) estimates on the unsupervised training set with varying Ω and ϵ and $N = 32000$. The values of κ and η are assumed to be known exactly.

d	$\Delta t = 2^{-8}$ μs	$\Delta t = 2^{-6}$ μs	$\Delta t = 2^{-4}$ μs
2000	1.12×10^{-2}, 8.01×10^{-4}	1.10×10^{-2}, 8.42×10^{-4}	1.23×10^{-2}, 2.12×10^{-3}
4000	5.87×10^{-3}, 4.56×10^{-4}	6.46×10^{-3}, 5.07×10^{-4}	6.39×10^{-3}, 1.68×10^{-3}
8000	3.08×10^{-3}, 2.32×10^{-4}	2.98×10^{-3}, 2.87×10^{-4}	3.34×10^{-3}, 1.46×10^{-3}
16 000	1.68×10^{-3}, 1.19×10^{-4}	1.11×10^{-3}, 1.63×10^{-4}	1.58×10^{-3}, 1.39×10^{-3}
∞	1.25×10^{-5}, 7.75×10^{-6}	1.76×10^{-5}, 7.07×10^{-5}	1.14×10^{-4}, 1.23×10^{-3}

the ground state. Note that while the choice of a single unknown parameter is made in this section to provide a clear and straightforward demonstration, the model still supports multiple possible parameters to be estimated in addition to the decoder degrees of freedom.

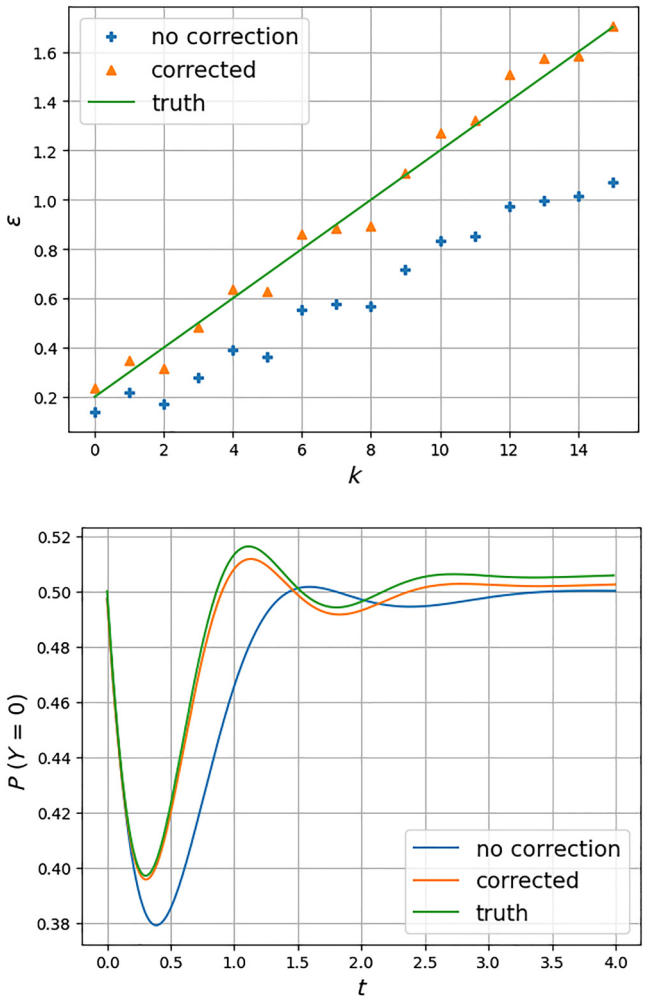

FIG. 6. Estimate of ϵ (top) and $P(Y_i = 0)$ when $\epsilon = 1.7$ (bottom) and measurement is in the X direction with spin-up in the Z direction as the initial state for both qubits for training data simulated with single-particle relaxation rate $\gamma_s = 0.1$ but a model that does not explicitly account for γ_s. Results are shown for the physical model alone and with a correction learned by the decoder LSTM.

In this case, we take a more-diverse set of measurements, simulating 10 000 trajectories measuring in each of the X, Y, and Z directions for both qubits, and relaxing κ to one fourth the value used in the last section to reduce measurement back action. Spin-up in the Z direction is the initial state for each qubit. This more-cautious approach to measurement is warranted if completely unknown effects are expected to be present. More information about the type of phenomenon, but not necessarily the magnitude, could allow a more-targeted measurement scheme, but here we keep it general. The trajectory group size d used for each input was 5000, and the best of 20 randomly initialized models was selected for the results in this section.

The results of the parameter estimation with and without the correction are shown in Fig. 6 and Table V. Here we see that single-particle relaxation has introduced a significant bias to the estimated ϵ parameters when unaccounted for in the model, but the decoder LSTM has successfully corrected for the effect, returning the MSE to a value much closer to where it would have been had the physical model explicitly accounted for it.

V. CONCLUSIONS

We have proposed a machine-learning model based on a denoising autoencoder capable of direct estimation of physical parameters in a system modeled by a stochastic master equation from weak-measurement records. The model is capable of learning in a supervised context and an unsupervised context, and it can accurately predict parameters for systems not seen in the training data. While leveraging the use of a master-equation integrator to enable unsupervised learning, the autoencoder is robust regarding

TABLE V. MSE of ϵ estimates obtained for different γ_s values in the model. The final column shows the results when learning is enabled for the free parameters in the decoder to account for the unanticipated term γ_s. The measurement-time spacing $\Delta t = 2^{-8}$ μs.

d	$\gamma_s = 0.1$	$\gamma_s = 0.0$	$\gamma_s = 0.0 +$ correction
5000	2.84×10^{-3}	0.142	3.59×10^{-3}
∞	2.12×10^{-6}	0.145	3.26×10^{-4}

unanticipated dynamics not included in its physical model. We have demonstrated this in the case of unanticipated Lindblad dissipative terms. Previous work established that LSTM models are capable of learning dynamics beyond unknown Lindbladian dissipation [15,21], and it will be an interesting subject for future work to examine the robustness of our model regarding the presence of unanticipated non-Markovian and nonlinear dynamics in the training data.

Another potential subject for future investigation is the ability of the model to estimate parameters for much-larger systems in the supervised context, where it may not necessarily be subject to the exponential scaling of the Hilbert space with system size that often limits other approaches to parameter estimation.

ACKNOWLEDGMENTS

This material is based on work supported by the National Science Foundation's Quantum Leap Big Idea under Grant No. OMA-1936388. A.K.R. and C.M. acknowledge support from the National Science Foundation's Quantum Leap Challenge Institute program under Grant No. OMA-2016244.

This work was performed, in part, at the Center for Integrated Nanotechnologies, an Office of Science User Facility operated for the U.S. Department of Energy (DOE) Office of Science.

This article has been co-authored by an employee of National Technology & Engineering Solutions of Sandia, LLC under Contract No. DE-NA0003525 with the U.S. Department of Energy (DOE). The employee owns all right, title and interest in and to the article and is solely responsible for its contents. The United States Government retains and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this article or allow others to do so, for United States Government purposes. The DOE will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan [53].

APPENDIX A: RECURRENT NEURAL NETWORKS

In this appendix, we describe RNNs in additional detail. Our presentation is mostly standard and closely follows the treatment in Ref. [54].

RNNs are a class of models which use a cycle within their network structure such that the output of a unit depends in some manner on its own output at some previous step, so they have an underlying recursive structure [44]. This structure allows RNNs to model sequences of data where information at the current index depends on information processed in the past. While the term "recurrent neural networks" refers to a general class of models that share this recursive pattern, in the ML literature, it is used to refer to a specific kind architecture with a simple cyclic pattern (sometimes also called "Elman networks") [55].

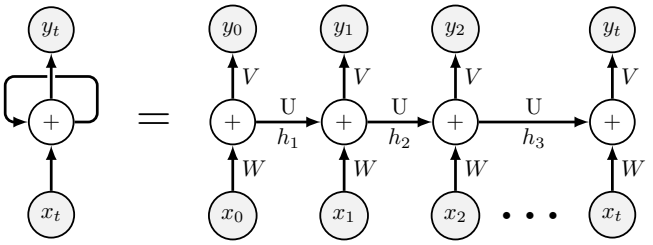

FIG. 7. Basic architecture of an RNN. The model on the left can be thought of as "unrolling" itself in time.

The model operates as follows (see Fig. 7). Given an input vector x_t, where t is an index over the sequence to be modeled, we first multiply x_t by a weight matrix W and pass the result through a nonlinearity g to produce values in a hidden layer h_t. However, unlike feed-forward networks, we supplement this value with values from previous hidden layers h_{t-1}, multiplied by a weight matrix U as well. This can be summarized as

$$h_t = g(Uh_{t-1} + Wx_t). \tag{A1}$$

Additionally, each unit of the RNN can produce an output y_t, which can be parameterized as follows:

$$y_t = f\,(Vh_t), \tag{A2}$$

where f is usually taken to be a softmax function and V is a weight matrix. This network is then trained through an analogue of backpropagation that unrolls the sequence along its index referred to as "backpropagation through time" [56–58].

In practice, it has been observed that such networks can be hard to train for longer sequence lengths. This is mainly due to the "bottleneck" caused by the previous hidden state vector h_{t-1} having to summarize *all* of the relevant information at the current index inside a fixed-size vector. Another difficulty encountered is due to repeated multiplications of small values in backpropagation through time causing poor gradient flow to steps further back in time (referred to as the "vanishing-gradient problem" [45]).

LSTMs are a commonly used extension of RNNs to tackle the shortcomings pointed out above [46]. The central idea is to gain more-precise control over information being allowed to flow through the network. In addition to the hidden state, an additional so-called memory cell (also called the "cell state") is computed, which, like the hidden state, allows the flow of information along the length of the

TABLE VI. MSEs for random-forest models with row-major input flattening and column-major input flattening, respectively.

d	Best MSE	Median MSE	Mean MSE
2000	2.35×10^{-3}, 2.63×10^{-3}	4.13×10^{-3}, 4.08×10^{-3}	4.41×10^{-3}, 4.52×10^{-3}
4000	1.23×10^{-3}, 1.31×10^{-3}	3.44×10^{-3}, 3.52×10^{-3}	3.69×10^{-3}, 3.70×10^{-3}
8000	7.49×10^{-4}, 7.34×10^{-4}	1.61×10^{-3}, 1.68×10^{-3}	1.86×10^{-3}, 1.86×10^{-3}
16 000	3.36×10^{-4}, 3.26×10^{-4}	1.47×10^{-3}, 1.50×10^{-3}	1.65×10^{-3}, 1.71×10^{-3}
∞	2.34×10^{-4}, 3.48×10^{-5}	1.54×10^{-3}, 1.37×10^{-3}	1.75×10^{-3}, 1.79×10^{-3}

sequence. The memory cell is designed such that information can be added to it or subtracted from it. See Ref. [30] for details of how this is done.

APPENDIX B: BASELINE—RANDOM FORESTS

To justify our choice of using LSTMs as the backbone of our proposed architecture, we provide a comparison using random forests [59] for the encoder-only experiments described in Sec. IV A with. We choose a comparison against random forests because they have been shown to have robust off-the-shelf performance across a wide variety of tasks [60]. In this appendix we give a brief overview of random forests and a comparison with results in Table I, which justify the use of neural network-based architectures in our subsequent experiments.

1. Model description

Here we provide a brief overview of the random-forest model, closely following the treatment in Ref. [61]. Decision trees are simple tree-based models that make decisions by recursively partitioning the input feature space into regions and then returning the label or output associated with that region. While decision trees are often robust regarding noisy features and exhibit a high degree of interpretability, they can grow to large depths, thereby overfitting the training set [61]. This effect is often addressed by learning a random forest, i.e., a collection (ensemble) of decision trees, each of which is constructed independently by training on a random subset of the data, in addition to using some random choices in the algorithm. This introduces diversity among the trees, preventing the model from becoming too dependent on any single feature. For regression tasks (i.e., the output is continuous valued), as in this work, the final prediction is typically the average of the predictions made by each tree.

The combination of random sampling and majority voting (or averaging) is a technique known as "bagging" (or "*b*ootstrap *agg*regat*ing*") [62]. Random forests use bagging to create an ensemble that is more robust (reducing uncertainty on the output) and less prone to overfitting compared than single decision tree. Apart from their versatility and robustness, random forests provide insights into feature importance, helping to identify the most-influential variables in the model.

2. Training setup

Our aim is to compare the prediction accuracy for ϵ for a fixed Ω in the case of supervised learning between the random-forest model and the model described in the main text. To do so, we use the same procedure as described in Sec. IV A. The training set contains averaged trajectories sampled randomly in groups of differing size d, and the trajectories in the dataset correspond to 40 values of ϵ evenly spaced on $[0, 2)$ for a fixed $\Omega = 1.395$ rad/μs. The crucial hyperparameters for our method are the number of trees in the ensemble (set to 100) and the maximum depth of each tree (set to 25). These were found via fivefold cross-validation over the dataset.

A clear point of difference compared with the LSTM-based model is the form of the input given to the random-forest model. Unlike the LSTM model, input to the random-forest model *must* be one dimensional. We use the concept of "dimensionality" to refer to the number of weak-measurement values at each time step. Our input is two dimensional since we have a weak-measurement value for each qubit. This dichotomy requires us to " flatten" the input values into a single dimension. This can

TABLE VII. MSE pair for ϵ estimates on the supervised training set with only the encoder used. The first value is the MSE of the random-forest model with column-major ordering, and the second value corresponds to the MSE of the LSTM-based model.

d	Best MSE	Median MSE	Mean MSE
2000	2.63×10^{-3}, 1.47×10^{-3}	4.08×10^{-3}, 1.51×10^{-3}	4.52×10^{-3}, 1.63×10^{-3}
4000	1.31×10^{-3}, 7.37×10^{-4}	3.52×10^{-3}, 7.16×10^{-4}	3.70×10^{-3}, 8.56×10^{-4}
8000	7.34×10^{-4}, 4.39×10^{-4}	1.68×10^{-3}, 6.00×10^{-4}	1.86×10^{-3}, 4.67×10^{-4}
16 000	3.26×10^{-4}, 1.83×10^{-4}	1.50×10^{-3}, 2.13×10^{-4}	1.71×10^{-3}, 2.68×10^{-4}
∞	3.48×10^{-5}, 8.11×10^{-6}	1.37×10^{-3}, 5.59×10^{-5}	1.79×10^{-3}, 7.41×10^{-5}

be done in two ways: either by keeping measurement values corresponding to a particular time step close together (column-major order) or by concatenating the values of the second qubit after *all* the values of the first qubit (row-major order). We find that the choice of ordering makes little difference to our results as we demonstrate next.

3. Results

We present the impact of the choice of input order in our random-forest model in Table VI. We find the difference in MSE values to be minimal.

We now compare the results of the random-forest model in column-major order with the results of the LSTM model in Table VII.

We find about an order-of-magnitude difference in the MSEs achieved by the random-forest model compared with the LSTM model. The trend of increased accuracy with increasing size of d, also observed in the LSTM results, continues to hold in this case. To get a better sense of the uncertainty, we also present the mean MSEs and their 1-standard-deviation error bars for both our models in Fig. 8. The fact that our choice of "flattening" does not affect the overall performance of the random-forest model suggests that the difference in performance between the random-forest model and the LSTM model could be attributed to the LSTM-based model taking the sequential nature of the data into account versus the random-forest model treating each trajectory as just a high-dimensional vector. We leave a thorough investigation of these issue to future work.

We also provide a comparison of the number of parameters used in both our models in Table VIII. Since the number of parameters in a random-forest model depends on the depth of the trees, which in turn depends on the complexity of the dataset, we report the maximum number of parameters across all models obtained after training.

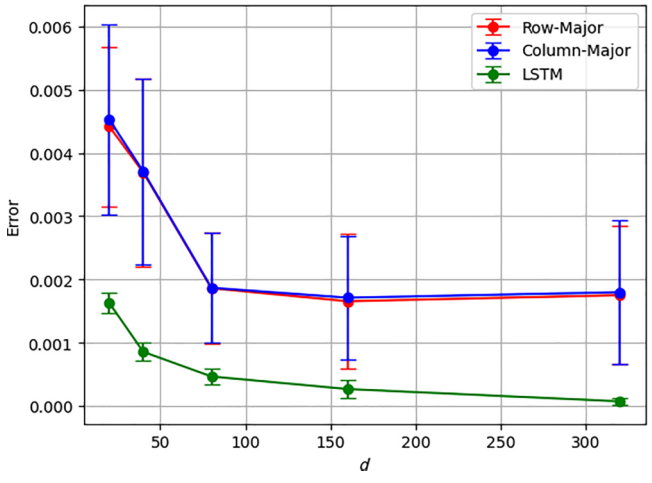

FIG. 8. Mean MSEs and their 1-standard-deviation error bars for the random-forest model with different input orderings and the LSTM model.

TABLE VIII. Number of parameters used in the two models studied.

Method	Number of parameters
LSTM	1 619 316
Random forest	129 380

[1] J. Preskill, Quantum computing in the NISQ era and beyond, Quantum **2**, 79 (2018).

[2] M. Sarovar, T. Proctor, K. Rudinger, K. Young, E. Nielsen, and R. Blume-Kohout, Detecting crosstalk errors in quantum information processors, Quantum **4**, 321 (2020).

[3] V. Tripathi, H. Chen, M. Khezri, K.-W. Yip, E. Levenson-Falk, and D. A. Lidar, Suppression of crosstalk in superconducting qubits using dynamical decoupling, Phys. Rev. Appl. **18**, 024068 (2022).

[4] A. Y. Kitaev, Quantum computations: Algorithms and error correction, Russ. Math. Surv. **52**, 1191 (1997).

[5] D. Aharonov and M. Ben-Or, in *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, STOC '97 (Association for Computing Machinery, New York, NY, USA, 1997), p. 176.

[6] J. Preskill, Reliable quantum computers, Proc. R. Soc. London. Ser. A: Math., Phys. Eng. Sci. **454**, 385 (1998).

[7] E. Knill, R. Laflamme, and W. H. Zurek, Resilient quantum computation, Science **279**, 342 (1998).

[8] D. Gottesman, Theory of fault-tolerant quantum computation, Phys. Rev. A **57**, 127 (1998).

[9] G. Carleo and M. Troyer, Solving the quantum many-body problem with artificial neural networks, Science **355**, 602 (2017).

[10] G. Carleo, Y. Nomura, and M. Imada, Constructing exact representations of quantum many-body systems with deep neural networks, Nat. Commun. **9**, 5322 (2018).

[11] J. Carrasquilla, G. Torlai, R. G. Melko, and L. Aolita, Reconstructing quantum states with generative models, Nat. Mach. Intell. **1**, 155 (2019).

[12] A. Rocchetto, E. Grant, S. Strelchuk, G. Carleo, and S. Severini, Learning hard quantum distributions with variational autoencoders, npj Quantum Inf. **4**, 28 (2018).

[13] J. Carrasquilla and G. Torlai, How to use neural networks to investigate quantum many-body physics, PRX Quantum **2**, 040201 (2021).

[14] G. Torlai, C. J. Wood, A. Acharya, G. Carleo, J. Carrasquilla, and L. Aolita, Quantum process tomography with unsupervised learning and tensor networks, Nat. Commun. **14**, 2858 (2023).

[15] L. Banchi, E. Grant, A. Rocchetto, and S. Severini, Modelling non-Markovian quantum processes with recurrent neural networks, New J. Phys. **20**, 123030 (2018).

[16] S. L. Brunton, J. L. Proctor, and J. N. Kutz, Discovering governing equations from data by sparse identification of nonlinear dynamical systems, Proc. Natl. Acad. Sci. **113**, 3932 (2016).

[17] K. Champion, B. Lusch, J. N. Kutz, and S. L. Brunton, Data-driven discovery of coordinates and governing equations, Proc. Natl. Acad. Sci. **116**, 22445 (2019).

[18] B. Lusch, J. N. Kutz, and S. L. Brunton, Deep learning for universal linear embeddings of nonlinear dynamics, Nat. Commun. **9**, 4950 (2018).

[19] E. Genois, J. A. Gross, A. Di Paolo, N. J. Stevenson, G. Koolstra, A. Hashim, I. Siddiqi, and A. Blais, Quantum-tailored machine-learning characterization of a superconducting qubit, PRX Quantum **2**, 040355 (2021).

[20] E. Flurin, L. S. Martin, S. Hacohen-Gourgy, and I. Siddiqi, Using a recurrent neural network to reconstruct quantum dynamics of a superconducting qubit from physical observations, Phys. Rev. X **10**, 011006 (2020).

[21] G. Koolstra, N. Stevenson, S. Barzili, L. Burns, K. Siva, S. Greenfield, W. Livingston, A. Hashim, R. K. Naik, J. M. Kreikebaum, K. P. O'Brien, D. I. Santiago, J. Dressel, and I. Siddiqi, Monitoring fast superconducting qubit dynamics using a neural network, Phys. Rev. X **12**, 031017 (2022).

[22] Z. An, J. Wu, M. Yang, D. L. Zhou, and B. Zeng, Unified quantum state tomography and hamiltonian learning: A language-translation-like approach for quantum systems, Phys. Rev. Appl. **21**, 014037 (2024).

[23] L. K. Castelano, I. Cunha, F. S. Luiz, R. de Jesus Napolitano, M. V. d. S. Prado, and F. F. Fanchini, Combining physics-informed neural networks with the freezing mechanism for general hamiltonian learning, Phys. Rev. A **110**, 032607 (2024).

[24] M. Raissi, P. Perdikaris, and G. Karniadakis, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, J. Comput. Phys 378, **686** (2019).

[25] R. T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud, in *Advances in Neural Information Processing Systems*, Vol. 31, edited by S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (Curran Associates, Inc., Montreal, Canada, 2018).

[26] M. Habiba and B. A. Pearlmutter, in *2020 31st Irish Signals and Systems Conference (ISSC)* (IEEE, Piscataway, NJ, Letterkenny, Ireland, 2020), p. 1.

[27] J. A. Smolin, J. M. Gambetta, and G. Smith, Efficient method for computing the maximum-likelihood quantum state from measurements with additive Gaussian noise, Phys. Rev. Lett. **108**, 070502 (2012).

[28] K. Kaheman, E. Kaiser, B. Strom, J. N. Kutz, and S. L. Brunton, Learning discrepancy models from experimental data, arXiv:1909.08574.

[29] B. M. de Silva, D. M. Higdon, S. L. Brunton, and J. N. Kutz, Discovery of physics from data: Universal laws and discrepancies, Front. Artif. Intell. **3**, 25 (2020).

[30] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning* (MIT Press, 2016), http://www.deeplearningbook.org.

[31] J. Haah, R. Kothari, and E. Tang, Learning quantum Hamiltonians from high-temperature Gibbs states and real-time evolutions, Nat. Phys. **20**, 1027 (2024).

[32] A. Anshu, S. Arunachalam, T. Kuwahara, and M. Soleimanifar, Sample-efficient learning of interacting quantum systems, Nat. Phys. **17**, 931 (2021).

[33] A. Anshu and S. Arunachalam, A survey on the complexity of learning quantum states, Nat. Rev. Phys. **6**, 59 (2024).

[34] H.-Y. Huang, Y. Tong, D. Fang, and Y. Su, Learning many-body Hamiltonians with Heisenberg-limited scaling, Phys. Rev. Lett. **130**, 200403 (2023).

[35] H. Li, Y. Tong, H. Ni, T. Gefen, *et al.*, Heisenberg-limited Hamiltonian learning for interacting bosons, npj Quantum Inf. **10**, 83 (2024).

[36] A. Dutkiewicz, T. E. O'Brien, and T. Schuster, The advantage of quantum control in many-body Hamiltonian learning, arXiv:2304.07172.

[37] E. Bairey, I. Arad, and N. H. Lindner, Learning a local Hamiltonian from local measurements, Phys. Rev. Lett. **122**, 020504 (2019).

[38] E. Bairey, C. Guo, D. Poletti, N. H. Lindner, and I. Arad, Learning the dynamics of open quantum systems from their steady states, New J. Phys. **22**, 032001 (2020).

[39] K. Jacobs and D. A. Steck, A straightforward introduction to continuous quantum measurement, Contemp. Phys. **47**, 279 (2006).

[40] T. A. Brun, A simple model of quantum trajectories, Am. J. Phys. **70**, 719 (2002).

[41] R. Bonifacio, P. Schwendimann, and F. Haake, Quantum statistical theory of superradiance. I, Phys. Rev. A **4**, 302 (1971).

[42] A. Blais, A. L. Grimsmo, S. M. Girvin, and A. Wallraff, Circuit quantum electrodynamics, Rev. Mod. Phys. **93**, 025005 (2021).

[43] J. Gambetta, A. Blais, M. Boissonneault, A. A. Houck, D. I. Schuster, and S. M. Girvin, Quantum trajectory approach to circuit QED: Quantum jumps and the Zeno effect, Phys. Rev. A **77**, 012112 (2008).

[44] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1: Foundations*, edited by D. E. Rumelhart and J. L. Mcclelland (MIT Press, Cambridge, MA, 1986), p. 318.

[45] S. Hochreiter, Y. Bengio, P. Frasconi, and J. Schmidhuber, in *A Field Guide to Dynamical Recurrent Neural Networks*, edited by S. C. Kremer and J. F. Kolen (IEEE Press, 2001).

[46] S. Hochreiter and J. Schmidhuber, Long short-term memory, Neural Comput. **9**, 1735 (1997).

[47] Y. Rubanova, R. T. Q. Chen, and D. K. Duvenaud, in *Advances in Neural Information Processing Systems*, Vol. 32, edited by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Curran Associates, Inc., Vancouver, BC, Canada, 2019).

[48] J. Haegeman, J. I. Cirac, T. J. Osborne, I. Pižorn, H. Verschelde, and F. Verstraete, Time-dependent variational principle for quantum lattices, Phys. Rev. Lett. **107**, 070601 (2011).

[49] J. Haegeman, C. Lubich, I. Oseledets, B. Vandereycken, and F. Verstraete, Unifying time evolution and optimization with matrix product states, Phys. Rev. B **94**, 165116 (2016).

[50] D. Perez-Garcia, F. Verstraete, M. M. Wolf, and J. I. Cirac, Matrix product state representations, Quantum Info. Comput. **7**, 401 (2007).

[51] U. Schollwöck, The density-matrix renormalization group in the age of matrix product states, Ann. Phys. (N. Y) **326**, 96 (2011).

[52] C. C. Bultink, B. Tarasinski, N. Haandbæk, S. Poletto, N. Haider, D. J. Michalak, A. Bruno, and L. DiCarlo, General method for extracting the quantum efficiency of dispersive qubit readout in circuit QED, Appl. Phys. Lett. **112**, 092601 (2018).

[53] https://www.energy.gov/downloads/doe-public-access-plan.
[54] D. Jurafsky and J. H. Martin, *Speech and Language Processing An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition* (2024), 3rd ed.
[55] J. L. Elman, Finding structure in time, Cogn. Sci. **14**, 179 (1990).
[56] A. Robinson and F. Fallside, *The Utility Driven Dynamic Error Propagation Network* (University of Cambridge Department of Engineering, 1987).
[57] P. J. Werbos, Generalization of backpropagation with application to a recurrent gas market model, Neural Netw. **1**, 339 (1988).
[58] M. C. Mozer, in *Backpropagation: Theory, Architectures, and Applications* (L. Erlbaum Associates Inc., USA, 1995), p. 137.
[59] L. Breiman, Random forests, Mach. Learn. **45**, 5 (2001).
[60] M. Fernández-Delgado, E. Cernadas, S. Barro, and D. Amorim, Do we need hundreds of classifiers to solve real world classification problems?, J. Mach. Learn. Res. **15**, 3133 (2014).
[61] T. Hastie, R. Tibshirani, and J. Friedman, in *Springer Series in Statistics* (Springer New York Inc., New York, NY, USA, 2001).
[62] L. Breiman, Bagging predictors, Mach. Learn. **24**, 123 (1996).