Model reductions in quantum optical devices

Edwin Ng, Tatsuhiro Onodera, and Gil Tabak
Edward L. Ginzton Laboratory, Stanford University, Stanford, CA 94305, USA
(Dated: December 11, 2015)

A major obstacle in designing quantum photonic systems is the exponential scaling in simulation complexity as a function of network size. On the other hand, in many applications, the individual components exhibit relatively simple, low-dimensional behavior within the parameter regimes of interest. In this project, we explore the use of machine learning to identify such structure and develop reduced models for efficient quantum simulation. We focus on the Kerr cavity device, for which a low-dimensional "qubit limit" exists under weak drive. Using PCA and manifold learning algorithms, we show that this analytic limit can indeed also be inferred from a learning approach.

I. INTRODUCTION

The state of any pure, finite-dimensional quantum system is described by a vector $\psi \in \mathbb{C}^N$, where $\|\psi\|_2^2 = 1$. While quantum optical systems are technically infinite-dimensional, as long as the energy of the system remains bounded, we can typically choose a suitable finite N to "truncate" the space (usually, $N \sim 10^2$ for each component). The dynamics of the quantum system are then generated by a (Hermitian) Hamiltonian matrix $H \in \mathbb{C}^{N\times N}$ along with a set of "collapse" operators $L = \{L_1, L_2, \ldots\}$, where each $L_i \in \mathbb{C}^{N\times N}$.

The most basic form of the Kerr cavity device features two parameters: a nonlinearity $\chi \in \mathbb{R}$, and a drive $\alpha \in \mathbb{R}$. A formal (SLH) description of this model can be found in [1], which includes how H and L depend on the parameters, but we omit the details here. We do note, however, that the qubit limit from [2] corresponds to a choice of parameters where $\chi \gg \alpha, 1$.

Once we construct H and L from the model, we can perform exact quantum simulations on the system (up to truncation errors). We use the Python-based package QuTIP [3], which provides two different solvers for quantum simulation. The first ("quantum trajectories") is a Monte Carlo solver that, given H, L, and an initial condition ψ_0, generates a (stochastic) time series of vectors $(\psi_1, \psi_2, \ldots, \psi_T)$, where for simplicity we have fixed some time step δ and run the simulation for a time $T\delta$. If we run M quantum trajectory simulations, then we obtain $S = \left\{\Psi^{(1)}, \ldots, \Psi^{(M)}\right\}$, where each $\Psi^{(i)} = \left(\psi_1^{(i)}, \ldots, \psi_T^{(i)}\right)$, so that $\psi_t^{(i)}$ is the state of the ith trajectory at time t. Quantum trajectories of this sort comprise the training data we use for machine learning.

The other solver provided by QuTIP is a "master equation" solver. While the quantum trajectories formalism solves the stochastic Schrödinger equation (which is a quantum stochastic differential equation [8]), the master equation is a corresponding partial differential equation describing the probability distribution over the ensemble of those trajectories. That is, if the trajectories ψ_t (as random variables) are drawn from some distribution $\mathcal{D}_t$, the master equation describes the time evolution of a "density matrix" $\rho_t = \mathrm{Ex}\left[\psi_t \psi_t^\dagger\right] \in \mathbb{C}^{N\times N}$. Simulating the master equation provides us with the "true distribution" of states in time, which we use in this project to validate the performance of our learning algorithms. Illustrations of both quantum trajectories and master equation simulations are shown in Figure 1.

Thus, provided some trajectory data S, we want to find manifolds $\Omega \subseteq \mathbb{C}^N$ of dimension $K < N$ to which the dynamics can be projected, while still producing states with distribution close to $\mathcal{D}_t$ in some appropriate sense. Generally, finding Ω is an unsupervised learning problem. In Section III, we use PCA to identify principal subspaces, allowing us to find linear projectors to construct a new $\widetilde{H}$ and $\widetilde{L}$ for the reduced model. In Section V, we use manifold learning to find more general Ω; the tradeoff, however, is we must transform the stochastic Schrödinger equation to obtain the reduced dynamics.

Furthermore, with an eye towards exploring the physics in a more general way, we also consider in Section IV the supervised learning problem where we augment S with features $X = \left\{x^{(1)}, \cdots, x^{(M)}\right\}$ (comprised of, say, different settings of the parameters χ, α, etc.), thus learning the "physical regimes" $\Omega(X)$.

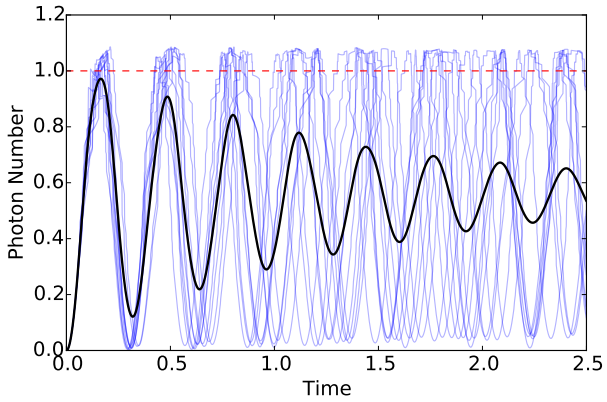

FIG. 1. Visualizing a set of $M = 10$ trajectories ($N = 10$, $\delta = 10^{-3}$, $\chi = -50$, $\alpha = 10$). To visualize $\psi_t^{(i)} \in \mathbb{C}^N$, we plot the "photon number" $n_t^{(i)} = \psi_t^{(i)\dagger} \,\mathrm{diag}(0, \ldots, N)\psi_t^{(i)}$; since $n_t^{(i)}$ does not exceed 1 (dashed line) by much, only the first two or three components of $\psi_t^{(i)}$ are important. The solid line shows $\mathrm{Ex}[n_t] = \mathrm{tr}[\rho_t \,\mathrm{diag}(0, \ldots, N)]$, with ρ_t obtained from master equation simulations.

II. RELATED WORK

To our knowledge, there are no explicitly machine learning-based methods for performing model reduction on the stochastic Schrödinger equation in particular. There is a body of literature on model reductions of the "stochastic master equation", which is yet another formulation of quantum dynamics useful for performing real-time coherent quantum feedback control. Many of these results tend to be analytic: for example, [4] assumes the quantum optical state is Gaussian, which works well in semi-classical regimes of operation. But this assumption, for example, does not hold for the system we are studying since a one-photon state is non-Gaussian.

In a more sophisticated approach, [5] uses a filtering framework to constrain the evolution of the quantum statistics (representing the state) to take place according to a predetermined, finite-dimensional family of probability densities. This latter approach is numerically efficient since the orthogonal projections can be analytically computed using the Fisher metric. However, the algorithm relies on the use of physical intuition to tailor the underlying family of densities to the physical system at hand, and as a result under-performs in some nonlinear problems such as absorptive bistability in cavity QED.

These limitations inspired [6] to use nonlinear manifold learning approaches for model reduction. In an approach similar to that in Section V, this work shows how to transform the stochastic master equation using a smoothed mapping into a lower-dimensional manifold obtained from a learning algorithm. However, the scheme is designed for active feedback control and hence integrates the stochastic master equation, requiring quadratically more memory than the stochastic Schrödinger equation we use in quantum simulations. As another key difference, [6] is concerned with stabilizing attractors in state space and hence works in a localized region of state space; consequently, it does not encounter global problems like manifolds folding on themselves, which we address in Section V by constructing multiple charts.

We do note that some work [7, 8] has been done on the simulation of the stochastic Schröndinger equation as well. Here, the main approach is to simulate the dynamics on a moving basis, optimizing for a coordinate transformation which centers the state in phase space before applying truncation. While this effectively treats systems with large displacement, it can under-perform when there is non-local entanglement in phase space (e.g., the infamous "Schrödinger cat" state).

In this context, we expect our approach of using machine learning from the very start can in general do better for quantum simulation applications, while making fewer assumptions about possible system behavior. Finally, we have also not found any prior work on the use of supervised learning to explore different regimes of operation for quantum optical devices. Therefore, we believe these techniques will prove to be a useful alternative for physicists to gain intuition about the systems under study.

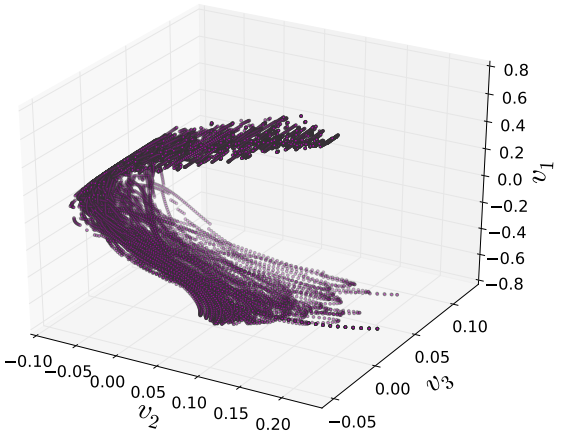

FIG. 2. The data in S plotted in the PCA basis using $K = 3$. Note that the variation in the data is much higher along v_1, which shows that the PCA is successful. However, we also see that the data shows additional nonlinear structure which is not captured by a linear model reduction algorithm like PCA.

III. IDENTIFYING PRINCIPAL SUBSPACES

Because of the linear structure of quantum mechanics, a natural reduction method is to find linear subspaces of $\mathbb{C}^N$ in which the dynamics lie. We use this as our starting point due to its simplicity and the fact that the qubit limit in [2] identifies precisely such a subspace. Because we want an orthonormal basis for the reduced space, we focus on principal components analysis (PCA).

We run PCA on an input set of MT real vectors

$$S = \left\{\psi_t^{(i)} : 1 \le t \le T \text{ and } 1 \le i \le M\right\},$$

where we have taken the absolute value of each component of the complex vectors: $\left(\psi_t^{(i)}\right)_j \mapsto \left|\left(\psi_t^{(i)}\right)_j\right|$. Applying PCA on S with produces N principal components $v_1, \ldots, v_N \in \mathbb{R}^N$, with corresponding variance (ratios) $w_1, \ldots, w_N$ such that $w_1 \ge \cdots \ge w_N$. These components form an orthogonal transformation $V = \begin{bmatrix} v_1 & \cdots & v_N \end{bmatrix}^{\mathrm{T}}$ from the original basis to a "PCA basis", where the state vectors from S written in the PCA basis have components with generally decreasing amplitude.

One way of performing a dimensional reduction using PCA is select a small set of principal components with non-negligible weight. For any $K \ge N$, we can define a dimensional reduction corresponding to truncating the last $N-K$ components in the PCA basis, represented by the $K \times N$ submatrix of V,

$$P = \begin{bmatrix} v_1 & v_2 & \cdots & v_K \end{bmatrix}^{\mathrm{T}}.$$

We think of P as a projection onto a K-dimensional subspace of $\mathbb{C}^N$, producing reduced-model simulation inputs $\widetilde{H} = PHP^\dagger$, $\widetilde{L} = \left\{PL_iP^\dagger : L_i \in L\right\}$, and $\widetilde{\psi}_0 = P\psi_0$. Then we say that the manifold found by the algorithm is the subspace $\Omega = \mathrm{span}(v_1, \cdots, v_K)$.

Variances from running PCA on S (same data as for Figure 1) are shown in the second column of Table I. In Figure 2, we illustrate some principal components of the data, in the PCA basis (that is, $P\psi$ for some subset of $\psi \in S$). Finally, Figure 3 illustrates the master equation simulation inaccuracies incurred by this model reduction for the qubit limit Kerr cavity given various choices of K. We see $K = 3$ already provides good approximation of the dynamics, as expected.

If we were to run quantum trajectories using the reduced simulation inputs $\widetilde{H}$, $\widetilde{L}$, and $\widetilde{\psi}_0$, we would obtain a set of trajectories $\widetilde{\psi}_t$ that converge to some density matrix $\widetilde{\rho}_t$ in expectation. We are interested in how close $P^\dagger \widetilde{\rho}_t P$ (i.e., in the original basis) is to the true density matrix ρ_t. While it is possible to compute an empirical estimate of $\widetilde{\rho}_t$ using reduced-model quantum trajectories, since our system is relatively simple ($N = 10$ for an exact quantum simulation) we can in fact run a reduced-model master equation simulation to find it directly and thus quantify the performance of our algorithm.

To define the error, we introduce the fidelity measure of quantum states [9], defined as

$$F(\rho, \sigma) = \mathrm{tr}\left(\sqrt{\rho^{1/2}\sigma\rho^{1/2}}\right),$$

which is well-defined for density matrices because they are positive semi-definite by construction. The fidelity is symmetric in its inputs and bounded between 0 and 1, approaching 1 when ρ and σ become indistinguishable (i.e., close in distribution). Thus, we define the error as

$$\varepsilon = 1 - \frac{1}{T}\sum_{t=1}^{T} F\left(\widetilde{\rho}_t, \rho_t\right).$$

For the data set S, the errors $\varepsilon(K)$ for different choices of K are shown in the third column of Table I. Again, we see that $K \sim 3$ is sufficient for achieving good performance.

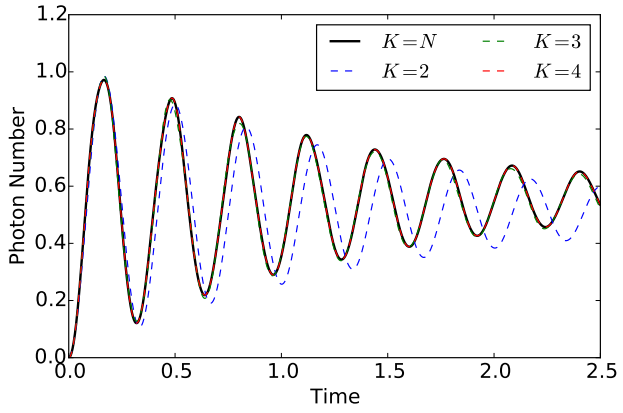

FIG. 3. Reduced-model master equation simulations obtained using PCA with $K = 2$, 3, and 4 principal components compared to the exact case of $K = N$. As in Figure 1, we visualize the state using the photon number. Notice that $K = 3$ and $K = 4$ are almost indistinguishable from full simulations.

Index k	PCA Variance w_k	Fid. Error $\varepsilon(K = k)$
1	9.63×10^{-1}	—
2	3.41×10^{-2}	6.54×10^{-2}
3	2.93×10^{-3}	1.47×10^{-3}
4	2.27×10^{-6}	6.63×10^{-6}
≥ 5	$< 10^{-9}$	$< 10^{-6}$

TABLE I. Summary of results for using PCA to find reduced models for qubit-limit Kerr cavity.

IV. LEARNING ON PRINCIPAL SUBSPACES

We next want to learn the principal subspace itself as a function of model parameters. To do this, we perform supervised learning using the parameters as features and the PCA projectors as targets. This means automating the procedure from Section III, then using a regression algorithm to learn the reduction. From the qubit limit results, we expect to find a low-dimensional (small K) subspace when $\chi \gg \alpha, 1$, with K increasing with α.

More formally, we have a supervised learning problem where we are provided $X = \left\{x^{(1)}, \ldots, x^{(M)}\right\}$, where each $x^{(i)} = \left(\chi^{(i)}, \alpha^{(i)}\right)$ for $\chi^{(i)}, \alpha^{(i)} \in \mathbb{R}$, augmented with $Z = \left\{S^{(1)}, \ldots, S^{(M)}\right\}$, where each $S^{(i)}$ is trajectory data for the parameter setting $\chi = \chi^{(i)}$ and $\alpha = \alpha^{(i)}$, consisting of $M_0 T$ N-dimensional vectors ($M_0 \ll M$ generally). We then compute for each $S^{(i)}$ a corresponding PCA decomposition $y^{(i)} = \left(w^{(i)}, V^{(i)}\right)$ where $w^{(i)} \in \mathbb{R}^N$ are the variances, and $V^{(i)} = \left(v_1^{(i)}, \ldots, v_N^{(i)}\right)$ are the PCA basis vectors. Then, X and $Y = \left\{y^{(1)}, \ldots, y^{(M)}\right\}$ form the input to the supervised learning problem.

For our training data, we sample a 10×10 uniform grid on the parameter intervals $4 \leq \chi \leq 14$ and $0.1 \leq \alpha \leq 7.5$. We then run $M_0 = 2$ trajectories for each parameter setting on this grid, keeping all other simulation settings the same as before. A minor issue is that the PCA vectors in Y occasionally "flip" due to the eigenvector subroutine; we address this issue by tracking the angles between nearest neighboring samples and correcting the flip.

To run the regression, we use a neutral network provided by the Python-based theanets package [10] to obtain regressions on the variances w and PCA basis vectors $v_1, \ldots, v_N$. More specifically, we train a neural network $\mathcal{N}_0$ on the mapping $x \mapsto w$, followed by N neural nets $\mathcal{N}_1, \ldots, \mathcal{N}_N$ on the mappings $x \mapsto v_1, \ldots, x \mapsto v_N$, respectively. The idea is to use $\mathcal{N}_0$ to predict the variances, then select K using a threshold tolerance. Once K is selected, we then construct the projector using the predictions of $\mathcal{N}_1, \ldots, \mathcal{N}_K$.

For $\mathcal{N}_0$, we use a single chain of layers with 2 input nodes (since $x \in \mathbb{R}^2$) and 10 output notes (since $w \in \mathbb{R}^{10}$), with $(10, 30)$ hidden layers. For each $\mathcal{N}_k$, we similarly use a single chain of layers with 2 input notes and 10 output notes (since each $v_k \in \mathbb{R}^{10}$), but with (8,15) hidden layers. Theanets uses the sigmoid thresholding function by default on its hidden layers.

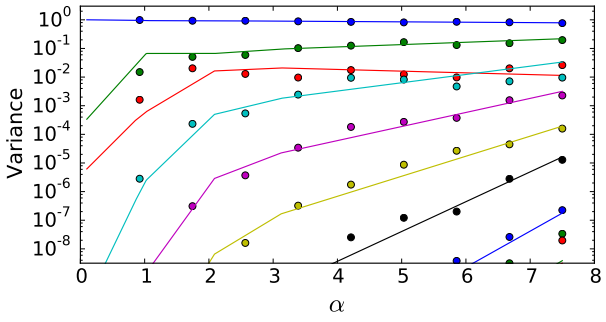

FIG. 4. Training and regressed variance ratios for neural network supervised learning, for the slice of $\chi = 4$. The points show the training variances $w^{(i)}$ corresponding to parameters $\alpha^{(i)}$, while the lines show the regressor's prediction at 100 points in the same interval. Note that the piecewise form of the lines come from the regressor rather than sampling.

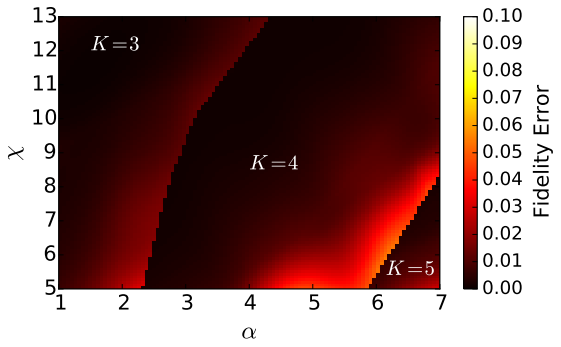

FIG. 5. Fidelity error of neural network regressor for learning PCA model reduction. The performance over the fine grid sweep (100×100) is quite smooth despite the training data being 10×10. The discontinuities in the error correspond precisely to jumps taken in K by the trained algorithm.

Having obtained the regression model, we can evaluate its performance by looking at the predictions made. For example, in Figure 4, we plot the training variances $w^{(i)}$ (as computed by PCA) alongside the regressed components of $w(\alpha)$ predicted by $\mathcal{N}_0$. We fix a tolerance $\epsilon_{\text{tol}} = 10^{-2}$, so that we drop all variances smaller than $\epsilon_{\text{tol}} w_1$, producing a set of K top principal components.

Next, from the regressors $\mathcal{N}_1, \ldots, \mathcal{N}_K$, we can obtain $P(\alpha, \chi) = \left[v_1(\alpha,\chi) \ \cdots \ v_K(\alpha,\chi)\right]^{\mathrm{T}}$. Generally, P will not be semi-orthogonal; we solve this issue by performing Gram-Schmidt orthgonalization on the rows. Finally, we truncate H and L to obtain a reduced model described by $\widetilde{H}(\alpha,\chi)$ and $\widetilde{L}(\alpha,\chi)$ with the procedure in Section III.

A heatmap showing the fidelity error of these regressed truncated models vs the true density matrix (again, via master equation simulations) is shown in Figure 5. As α increases, the neural network realizes the need to taking increasing values of K, while $K = 3$ when $\alpha \ll \chi$, as expected from the qubit limit. The presence and shape of these "regime transitions" can provide substantial insight into the physics (e.g., how χ and α scale, etc.).

V. NONLINEAR EMBEDDINGS

Despite the utility of PCA in identifying linear subspaces for model reduction, quantum dynamics tend to lie on *nonlinear* manifolds in general (e.g., see Figure 2), so linear reductions can fail to both find optimal values for K and preserve low error rates. For this reason, we next consider nonlinear approaches for model reduction. Here, the goal is to find a smooth map from $\mathbb{C}^N$ to a parameterized K-dimensional manifold Ω on which the dynamics lie. Simulation of the system on Ω is then expected to be more efficient than in $\mathbb{C}^N$.

In this project, we explored different methods of nonlinear manifold learning such as variations of local linear embedding (LLE) and local tangent space alignment (LTSA). Since manifold learning algorithms work on real vectors only, we use the same trajectory data as in Section III, but preprocess it to obtain a set of MT real $2N$-dimensional vectors $X = \{(\mathrm{Re}\,\psi, \mathrm{Im}\,\psi) : \psi \in S\}$.

Generally, we found LTSA produces results that best preserved the visual structure of the data, so we focus on this algorithm here. Briefly, LTSA computes local tangent spaces of dimension K to the data and tries to align neighboring tangent spaces as it proceeds, thus producing a (sampled) coordinate parameterization of the K-dimensional manifold underlying the data.

As shown in Figure 6, the manifold obtained by LTSA appears to be 2-dimensional but folds over itself, suggesting the need for multiple coordinate charts to capture the structure. We introduce such coordinate charts $g_1, ..., g_C$, with each g_j a mapping $S_j \to \mathbb{R}^K$, where $S_j \subseteq \mathbb{R}^{2N}$ such that $\cup_j S_j = \mathbb{R}^{2N}$. For this particular training set, we define $S_j = \{x : x_j > \epsilon\}, S_{C/2+j} = \{x : x_j \leq \epsilon\}$ for $j = 1, \ldots, C/2$. Using $\epsilon = 0.4$, we illustrate two charts in Figure 6.

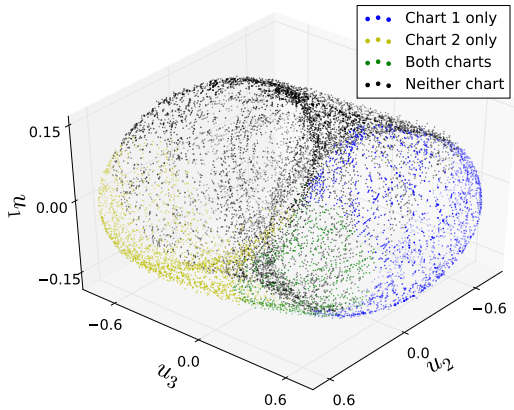

FIG. 6. LTSA output points for $K = 3$. A two-dimensional surface is discernible, but the topology is different from $\mathbb{R}^2$; multiple coordinate charts are needed. We illustrate two such charts here: points belonging to only either chart are in blue or yellow, points belonging to both in green.

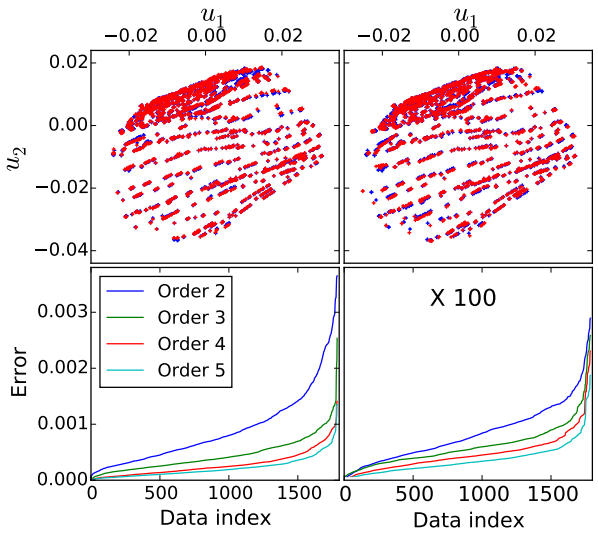

FIG. 7. Top: Comparison of LTSA coordinates for X (blue) with $\hat{g}_1$ (left) and $\hat{g}_1 \circ \hat{f}_1 \circ \hat{g}_1$ (right) evaluated on X, for order 4 polynomials. Bottom: Sorted L^2 errors of above points for polynomials of different orders. Note the errors on the right are multiplied by 100.

In addition to the charts g_j, we also require a corresponding inverse f_j, such that $f_j(g_j(x)) = x$ for all $x \in S_j$. Now, LTSA only gives numerical values of g_j sampled X. In order to obtain approximate charts $\hat{g}_j$ and their inverses $\hat{f}_j$, then, we need to solve a regression problem. From a number of regression algorithms tried, the best seems to be ridge (i.e., regularized L^2) regression. For simplicity, we use low-order polynomials for fitting both $\hat{g}_i$ and $\hat{f}_i$. Cross-validation results (using a 70%/30% split) are shown in Figure 7.

Ideally, the maps $\hat{f}_j$ and $\hat{g}_j$ should have as few parameters as possible (i.e., perform feature selection). From the regression above, we find that the vast majority of coefficients are small ($< 10^{-10}$); removing these coefficients leave approximately 200 significant ones. To see how removing more points would affect the regressed maps, we show in Figure 8 the errors incurred on the training data for various procedures for feature selection.

A key difficulty incurred by using nonlinear manifolds for model reduction is that we need to perform a corresponding nonlinear transformation on the stochastic Schrödinger equation in order to perform quantum simulations. Briefly, this SDE is given in general by

$$\mathrm{d}\psi = \mu(\psi)\,\mathrm{d}t + \sigma(\psi)\,\mathrm{d}\xi,$$

where $\mathrm{d}\xi$ is a complex vector-valued Wiener noise process, $\mu(\psi) = -\mathrm{i}H\psi - \frac{1}{2}\sum_{L_i \in L}\left(L_i^\dagger L_i - 2\ell_i^* L_i + \ell_i^* \ell_i\right)\psi$, and $\sigma_{ij}(\psi) = (L_j - \ell_j)\psi_i$; here, $\ell_i = \psi^\dagger L_i \psi$. In our formulation in terms of real vectors, we can simply double the dimensionality and can easily derive a similar real-valued SDE with corresponding terms μ' and σ', taking $\xi \mapsto W$ for a real vector-valued Wiener noise process.

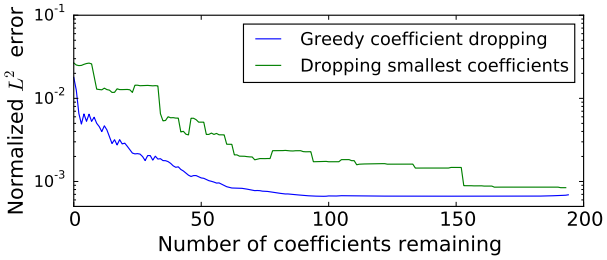

FIG. 8. Normalized L^2 error between LTSA coordinates for X and $\hat{g}_1$ evaluated on X.

Now we can perform model reduction by applying the Itô's lemma to find the evolution of the *coordinates*. For each chart g_j, we have a set of coupled reduced SDEs (superscripts denote components)

$$\mathrm{d}u^k = \mathrm{d}g^k(u) = \widetilde{\mu}_j(u)\,\mathrm{d}t + \widetilde{\sigma}_j(u)\,\mathrm{d}W,$$

where by Itô's lemma [11],

$$\widetilde{\mu}_j = \left[(\nabla g_j^k)^{\mathrm{T}}\mu'\right] \circ f_j + \frac{1}{2}\,\mathrm{tr}\left[{\sigma'}^{\mathrm{T}}\left(\mathrm{H}g_j^k\right)\sigma'\right] \circ f_j$$
$$\widetilde{\sigma}_j = \left[(\nabla g_j^k)^{\mathrm{T}}\sigma'\right] \circ f_j$$

where ∇ is the gradient and H is the Hessian. Since this results in a standard real-valued SDE, we can simulate this system to obtain the reduced dynamics.

One remaining subtlety is that we need to know which chart to use for each integration step, since we need to pick the right set of SDEs to use. Suppose we have a point u_t on chart j_t, and we wish to compute u_{t+1}. One possible method for selecting the chart is to maximize the minimum distance from all points in that chart:

$$j_{t+1} = \arg\max_j \left(\min_{s \in S_j} \|s - f_{j_t}(u_t)\|\right).$$

This concludes a possible algorithm for performing model reduction on nonlinear manifolds.

VI. CONCLUSION

In this project, we found wide applicability of machine learning methods for model reduction in quantum optical simulations. Using PCA, we implemented linear truncations into principal subspaces, which performed well as measured by low fidelity error. Subsequently, we extended this result into a supervised learning problem, allowing us to systematically identify different regimes of physics (the Kerr qubit limit, etc.). Finally, we explored reductions using nonlinear manifolds with multiple charts to perform model reduction for the stochastic Schrödinger equation. We derived the transformed equations of motion and devised a prescription for moving between charts. Looking ahead, we would like to implement this procedure in the near future, verify its accuracy in simulation, and apply the supervised learning approach to nonlinear reductions as well.

[1] See the IPython notebook http://web.stanford.edu/~edwin98/cs229-project/KERR-Qutip.html.

[2] H. Mabuchi, "Qubit limit of cavity nonlinear optics", Phys. Rev. A **85** 015806 (2012). doi:10.1103/PhysRevA.85.015806

[3] Documentation available at http://qutip.org/docs/3.1.0/index.html.

[4] D.A. Steck, K. Jacobs, H. Mabuchi, S. Habib, and T. Bhattacharya, "Feedback cooling of atomic motion in cavity QED", Phys. Rev. A **74**, 012322 (2006). doi:10.1103/PhysRevA.74.012322

[5] R. van Handel and H. Mabuchi, "Quantum projection filter for a highly nonlinear model in cavity QED", J. Opt. B **7** S226–S236 (2005). doi:10.1088/1464-4266/7/10/005

[6] A.E.B. Nielsen, A.S. Hopkins, and H. Mabuchi, "Quantum filter reduction for measurement-feedback control via unsupervised manifold learning", New J. Phys. **11** 105043 (2009). doi:10.1088/1367-2630/11/10/105043

[7] N. Tezak, N.H. Amini, and H. Mabuchi, "Quantum information geometry and localized quantum dynamics" (manuscript under preparation).

[8] T. Steimle, G. Alber, and I.C. Percival, "Mixed classical-quantal representation for open quantum systems" J. Phys. A **28**, L491 (1999). doi:10.1088/0305-4470/28/18/003

[9] M.A. Nielsen and I.L. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, 2010), Sec. 9.2.2.

[10] Documentation available at http://theanets.readthedocs.org/en/stable/.

[11] B. Øksendal, *Stochastic Differential Equations* (Springer Berlin Heidelberg, 2003).