Learning Hidden Quantum Markov Models

Siddarth Srinivasan
Georgia Institute of Technology

Geoff Gordon
Carnegie Mellon University

Byron Boots
Georgia Institute of Technology

Abstract

Hidden Quantum Markov Models (HQMMs) can be thought of as quantum probabilistic graphical models that can model sequential data. We extend previous work on HQMMs with three contributions: (1) we show how classical hidden Markov models (HMMs) can be simulated on a quantum circuit, (2) we reformulate HQMMs by relaxing the constraints for modeling HMMs on quantum circuits, and (3) we present a learning algorithm to estimate the parameters of an HQMM from data. While our algorithm requires further optimization to handle larger datasets, we are able to evaluate our algorithm using several synthetic datasets generated by valid HQMMs. We show that our algorithm learns HQMMs with the same number of hidden states and predictive accuracy as the HQMMs that generated the data, while HMMs learned with the Baum-Welch algorithm require more states to match the predictive accuracy.

1 INTRODUCTION

We extend previous work on Hidden Quantum Markov Models (HQMMs), and propose a novel approach to learning these models from data. HQMMs can be thought of as a new, expressive class of graphical models that have adopted the mathematical formalism for reasoning about uncertainty from quantum mechanics. We stress that while HQMMs could naturally be implemented on quantum computers, we do not need such a machine for these models to be of value. Instead, HQMMs can be viewed as novel models inspired by quantum mechanics that can be run on classical computers. In considering these models, we are interested in answering three questions: (1) how can we construct quantum circuits to simulate classical Hidden Markov Models (HMMs); (2) what happens if we take full advantage of this quantum circuit instead of enforcing the classical probabilistic constraints; and (3) how do we learn the parameters for quantum models from data?

The paper is structured as follows: first we describe related work and provide background on quantum information theory as it relates to our work. Next, we describe the hidden quantum Markov model and compare our approach to previous work in detail, and give a scheme for writing *any* hidden Markov model as an HQMM. Finally, our main contribution is the introduction of a maximum-likelihood-based unsupervised learning algorithm that can estimate the parameters of an HQMM from data. Our implementation is slow to train HQMMs on large datasets, and will require further optimization. Instead, we evaluate our learning algorithm for HQMMs on several simple synthetic datasets by learning a quantum model from data and filtering and predicting with the learned model. We also compare our model and learning algorithm to maximum likelihood for learning hidden Markov models and show that the more expressive HQMM can match HMMs' predictive capability with fewer hidden states on data generated by HQMMs.

2 BACKGROUND

2.1 Related Work

Hidden Quantum Markov Models were introduced by Monras et al. [2010], who discussed their relationship to classical HMMs, and parameterized these HQMMs using a set of Kraus operators. Clark et al. [2015] further investigated HQMMs, and showed that they could be viewed as open quantum systems with instantaneous feedback. We arrive at the same Kraus operator representation by building a quantum circuit to simulate a classical HMM and then relaxing some constraints.

Our work can be viewed as extending previous work by Zhao and Jaeger [2010] on Norm-observable operator models (NOOM) and Jaeger [2000] on observable-operator models (OOM). We show that HQMMs can be viewed as complex-valued extensions of NOOMs, formulated in the language of quantum mechanics. We

Proceedings of the 21^{st} International Conference on Artificial Intelligence and Statistics (AISTATS) 2018, Lanzarote, Spain. PMLR: Volume 84.

use this connection to adapt the learning algorithm for NOOMs in Zhao and Jaeger [2007] into the first known learning algorithm for HQMMs, and demonstrate that the theoretical advantages of HQMMs also hold in practice.

Schuld et al. [2015a] and Biamonte et al. [2016] provide general overviews of quantum machine learning, and describe relevant work on HQMMs. They suggest that developing algorithms that can learn HQMMs from data is an important open problem. We provide just such a learning algorithm in Section 4.

Other work at the intersection of machine learning and quantum mechanics includes Wiebe et al. [2016] on quantum perceptron models and learning algorithms. Schuld et al. [2015b] discuss simulating a perceptron on a quantum computer.

2.2 Belief States and Quantum States

Classical discrete latent variable models represent uncertainty with a probability distribution using a vector $\vec{x}$ whose entries describe the probability of being in the corresponding system state. Each entry is real and non-negative, and the entries sum to 1. In general, we refer to the run-time system component that maintains a state estimate of the latent variable as an 'observer', and we refer to the observer's state as a 'belief state.'

In quantum mechanics, the quantum state of a particle A can be written using Dirac notation as $|\psi\rangle_A$, a column-vector in some orthonormal basis (the row-vector is the complex-conjugate transpose ${}_A\langle\psi| = (|\psi\rangle_A)^\dagger$) with each entry being the 'probability amplitude' corresponding to that system state. The squared norm of the probability amplitude for a system state is the probability of observing that state, so the sum of squared norms of probability amplitudes over all the system states must be 1 to conserve probability. For example, $|\psi\rangle = \begin{bmatrix}\frac{1}{\sqrt{2}} & \frac{-i}{\sqrt{2}}\end{bmatrix}^\dagger$ is a valid quantum state, with basis states 0 and 1 having equal probability $\left\|\frac{1}{\sqrt{2}}\right\|^2 = \left\|\frac{i}{\sqrt{2}}\right\|^2 = \frac{1}{2}$. However, unlike classical belief states such as $\vec{x} = \begin{bmatrix}\frac{1}{2} & \frac{1}{2}\end{bmatrix}^T$, where the probability of different states reflects ignorance about the underlying system, a pure quantum state like the one described above is the *true* description of the system.

But how can we describe classical mixtures of quantum systems ('mixed states'), where we also have classical uncertainty about the underlying quantum states? Such information can be captured by a 'density matrix.' Given a mixture of N quantum systems, each with probability p_i, the density matrix for this ensemble is defined as follows:

$$\hat{\rho} = \sum_i^N p_i |\psi_i\rangle\langle\psi_i| \tag{1}$$

The density matrix is the general quantum equivalent of the classical belief state $\vec{x}$ and has diagonal elements representing the probabilities of being in each system state. Consequently, the normalization condition is $\mathrm{tr}(\hat{\rho}) = 1$. The off-diagonal elements represent quantum coherences, which have no classical interpretation. The density matrix $\hat{\rho}$ is Hermitian and can be used to describe the state of any quantum system.

The density matrix can also be extended to represent the joint state of multiple variables, or that of 'multi-particle' systems, to use the physical interpretation. If we have density matrices $\hat{\rho}_A$ and $\hat{\rho}_B$ for two qudits (a d-state quantum system, akin to qubits or 'quantum bits' which are 2-state quantum systems) A and B, we can take the tensor product to arrive at the density matrix for the joint state of the particles, as $\hat{\rho}_{\mathrm{AB}} = \hat{\rho}_A \otimes \hat{\rho}_B$. As a valid density matrix, the diagonal elements of this joint density matrix represent probabilities; $\mathrm{tr}\left(\hat{\rho}_{\mathrm{AB}}\right) = 1$, and the probabilities correspond to the states in the Cartesian product of the basis states of the composite particles. In this paper, the joint density matrix will serve as the analogue to classical joint probability distribution, with the off-diagonal terms encoding extra 'quantum' information.

Given the joint state of a multi-particle system, we can examine the state of just one or few of the particles using the 'partial trace' operation, where we 'trace over' the particles we wish to disregard. This lets us recover a 'reduced density matrix' for a subsystem of interest. The partial trace for a two-particle system $\hat{\rho}_{AB}$ where we trace over the second particle to obtain the state of the first particle is:

$$\hat{\rho}_A = \mathrm{tr}_B\left(\hat{\rho}_{AB}\right) = \sum_j {}_B\langle j|\hat{\rho}_{AB}|j\rangle_B \tag{2}$$

For our purposes, this operation will serve as the quantum analogue of classical marginalization.

Finally, we discuss the quantum analogue of 'conditioning' on an observation. In quantum mechanics, the act of measuring a quantum system can change the underlying distribution, i.e., collapses it to the observed state in the measurement basis, and this is represented mathematically by applying von Neumann projection operators (denoted $\hat{P}_y$ in this paper) to density matrices describing the system. One can think of the projection operator as having ones in the diagonal entries corresponding to observed system states and zeros elsewhere. If we only observe part of a larger system, the system collapses to the states where that subsystem had the observed result. For example, suppose we have the following density matrix for a two-state two-particle system with basis $\{|0\rangle_A|0\rangle_B, |0\rangle_A|1\rangle_B, |1\rangle_A|0\rangle_B, |1\rangle_A|1\rangle_B\}$:

$$\hat{\rho}_{AB} = \begin{bmatrix} 0.25 & 0 & 0 & 0 \\ 0 & 0.25 & -0.5 & 0 \\ 0 & -0.5 & 0.25 & 0 \\ 0 & 0 & 0 & 0.25 \end{bmatrix} \tag{3}$$

Table 1: Comparison between classical and quantum representations

Classical probability		**Quantum Analogue**	
Description	Representation	Representation	Description
Belief State	$\vec{x}$	$\hat{\rho}$	Density Matrix
Joint Distribution	$\vec{x}_1 \otimes \vec{x}_2$	$\hat{\rho}_{X_1} \otimes \hat{\rho}_{X_2}$	Multi-particle Density Matrix
Marginalization	$\vec{x} = \sum_y (\vec{x} \otimes \vec{y})$	$\hat{\rho} = \mathrm{tr}_Y(\hat{\rho}_X \otimes \hat{\rho}_Y)$	Partial Trace
Conditional probability	$P(\vec{x}\|y) = \frac{P(y,\vec{x})}{P(y)}$	$P(\text{states}\,\|y) \propto \mathrm{tr}_Y(\hat{P}_y \hat{\rho}_{XY} \hat{P}_y^\dagger)$	Projection + Partial Trace

Suppose we measure the state of particle B, and find it to be in state $|1\rangle_B$. The corresponding projection operator is $\hat{P}_{1_B} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$ and the collapsed state is now: $\hat{\rho}_{AB} = \hat{P}_{1_B} \hat{\rho}_{AB} \hat{P}_{1_B}^\dagger \overset{normalize}{\longrightarrow} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.5 \end{bmatrix}$. When we trace over particle A to get the state of particle B, the result is $\hat{\rho}_B = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$, reflecting the fact that particle B is now in state $|1\rangle_B$ with certainty. Tracing over particle B, we find $\hat{\rho}_A = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}$, indicating that particle A still has an equal probability of being in either state. Note that the underlying distribution of the system $\hat{\rho}_{AB}$ has changed; the probability of measuring the state of particle B to be $|0\rangle_B$ is now 0, whereas before measurement we had a $0.25 + 0.25 = 0.5$ chance of measuring $|0\rangle_B$. This is unlike classical probability where measuring a variable doesn't change the underlying distribution. We will use this fact when we construct our quantum circuit to simulate HMMs.

Thus, if we have an n-state quantum system that tracks a particle's evolution, and an s-state quantum system that tracks the likelihood of observing various outputs as they depend (probabilistically) on the n-state system, to obtain the n-state system conditioned on observation y, we apply the projection operator $\hat{P}_y$ on the joint system and trace over the second particle.

2.3 Hidden Markov Models

Classical Hidden Markov Models (HMMs) are graphical models used to model dynamic processes that exhibit Markovian state evolution. Figure 1 depicts a classical HMM, where the transition matrix $\mathbf{A}$ and emission matrix $\mathbf{C}$ are column-stochastic matrices that determine the Markovian hidden state-evolution and observation probabilities respectively. Bayesian inference can be used to track the evolution of the hidden variable.

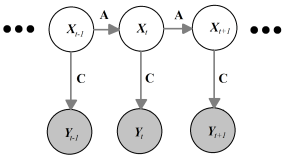

Figure 1: Hidden Markov Model

The belief state at time t is a probability distribution over states, and prior to any observation is written as:

$$\vec{x}'_t = \mathbf{A}\vec{x}_{t-1} \tag{4}$$

The probabilities of observing each output at time t is given by the vector $\vec{s}$:

$$\vec{s}_t = \mathbf{C}\vec{x}'_t = \mathbf{C}\mathbf{A}\vec{x}_{t-1} \tag{5}$$

We can use Bayesian inference to write the belief state vector after conditioning on observation y:

$$\vec{x}_t = \frac{\mathrm{diag}(\mathbf{C}_{(y,:)})\mathbf{A}\vec{x}_{t-1}}{\mathbb{1}^T \mathrm{diag}(\mathbf{C}_{(y,:)})\mathbf{A}\vec{x}_{t-1}} \tag{6}$$

where $\mathrm{diag}(\mathbf{C}_{(y,:)})$ is a diagonal matrix with the entries of the yth row of $\mathbf{C}$ along the diagonal, and the denominator renormalizes the vector $\vec{x}_t$.

An alternate representation of the Hidden Markov Model uses 'observable' operators (Jaeger [2000]). Instead of using the matrices $\mathbf{A}$ and $\mathbf{C}$, we can write $\mathbf{T}_y = \mathrm{diag}(\mathbf{C}_{(y,:)})\mathbf{A}$. There is a different operator $\mathbf{T}_y$ for each possible observable output y and $[\mathbf{T}_y]_{ij} = P(y; i_t | j_{t-1})$. We can then rewrite Equation 6 as:

$$\vec{x}_t = \frac{\mathbf{T}_y \vec{x}_{t-1}}{\mathbb{1}^T \mathbf{T}_y \vec{x}_{t-1}} \tag{4}$$

If we observe outputs $y_1, \ldots, y_n$, we apply $\mathbf{T}_n \ldots \mathbf{T}_1 \vec{x}$ and take the sum of the resulting vector to find the probability of observing the sequence, or renormalize to find the belief state after the final observation.

3 HIDDEN QUANTUM MARKOV MODELS

3.1 A Quantum Circuit to Simulate HMMs

Let us now contrast state evolution in quantum systems with state evolution in HMMs. The quantum analogue of observable operators is a set of non-trace-increasing Kraus operators $\{\hat{K}_i\}$ that are completely positive (CP) linear maps. Trace-preserving Kraus operators $\sum_i^N \hat{K}_i^\dagger \hat{K}_i = \mathbb{I}$, can map a density operator to another density operator. Trace-decreasing Kraus operators $\sum_i^N \hat{K}_i^\dagger \hat{K}_i < \mathbb{I}$, represent operations on a smaller part of a quantum system that can allow probability to 'leak' to other states that aren't being considered. This paper will formulate problems such that all sets of Kraus operators are trace-preserving. When there is only one operator in the set, i.e., $\hat{U}$ such that $\hat{U}^\dagger \hat{U} = \mathbb{I}$, then $\hat{U}$ is a unitary matrix. Unitary operators generally model the evolution of the 'whole' system, which may be high-dimensional. But if we care only about tracking the evolution of a smaller sub-system, which may interact

(a) Full Quantum Circuit to implement HMM

(b) Simplified scheme to implement HMM

Figure 2: HMM implementation on quantum circuits

(a) HQMM scheme with separate transition and emission; $\mathcal{K}_w = \sum_w \hat{K}_w(\cdot)\hat{K}_w^\dagger$

(b) A generalized scheme for HQMMs; $\mathcal{K}_{w,y_{t-1}} = \sum_w \hat{K}_{w,y_{t-1}}(\cdot)\hat{K}_{w,y_{t-1}}^\dagger$

Figure 3: Quantum schemes implementing classical HMMs

with its environment, we can use Kraus operators. The most general quantum operation that can be performed on a density matrix is $\hat{\rho}' = \frac{\sum_i^M K_i^\dagger \hat{\rho} K_i}{\mathrm{tr}\left(\sum_i^M K_i^\dagger \hat{\rho} K_i\right)}$, where the denominator re-normalizes the density matrix.

Now, how do we simulate classical HMMs on quantum circuits with qudits, where computation is done using unitary operations? There is no general way to convert column-stochastic transition and emission matrices to unitary matrices, so we prepare 'ancilla' particles and construct unitary matrices (see Algorithm 3 in the appendix for details) to act on the joint state. We then trace over one particle to obtain the state of the other.

Figure 2a illustrates a quantum circuit constructed with these unitary matrices. We prepare the 'ancilla' states $\hat{\rho}_{X_t}$ and $\hat{\rho}_{Y_t}$ appropriately (i.e., entirely in system state 1, represented by a density matrix of zeros except $\hat{\rho}_{1,1} = 1$), and construct $\hat{U}_1$ and $\hat{U}_2$ from transition matrix $\mathbf{A}$ and emission matrix $\mathbf{C}$, respectively. $\hat{U}_1$ evolves $(\hat{\rho}_{t-1} \otimes \hat{\rho}_{X_t})$ to perform Markovian transition, while $\hat{U}_2$ updates $\hat{\rho}_{Y_t}$ to contain the probabilities of measuring each observable output. At runtime, we measure $\hat{\rho}_{Y_t}$ which changes the joint distribution of $\hat{\rho}_{X_t} \otimes \hat{\rho}_{Y_t}$ to give the updated conditioned state $\hat{\rho}_t$. Mathematically, this is equivalent to applying a projection operator on the joint state and tracing over $\hat{\rho}_{Y_t}$. Thus, the forward algorithm that explicitly models a hidden Markov Model on a quantum circuit as per Figure 2a is written as:

$$\hat{\rho}_t \propto \mathrm{tr}_{\hat{\rho}_{Y_t}}\left(\hat{P}_y \hat{U}_2 \left(\mathrm{tr}_{\hat{\rho}_{t-1}}(\hat{U}_1(\hat{\rho}_{t-1} \otimes \hat{\rho}_{X_t})\hat{U}_1^\dagger) \otimes \hat{\rho}_{Y_t}\right) \hat{U}_2^\dagger \hat{P}_y^\dagger\right) \tag{7}$$

We can simplify this circuit to use Kraus operators acting on the lower-dimensional state space of $\hat{\rho}_{X_t}$. Since we always prepare $\hat{\rho}_{Y_t}$ in the same state, the operation $\hat{U}_2$ on the joint state of $\hat{\rho}_{X_t} \otimes \hat{\rho}_{Y_t}$ followed by the application of the projection operator $\hat{P}_y$ can be more concisely written as a Kraus operator on just $\hat{\rho}_{X_t}$, so that we need only be concerned with representing how the particle $\hat{\rho}_{X_t}$ evolves. We would need to construct a set of Kraus operators $\{\hat{K}_y\}$ for each observable output y, such that $\sum_y (\hat{K}_y)^\dagger (\hat{K}_y) = \mathbb{I}$.

Tensoring with an ancilla qudit and tracing over a qudit can be achieved with an $ns \times n$ matrix W and an $n \times ns$ matrix V_y respectively, since we always prepare our ancilla qudits in the same state (details on constructing these matrices can be found in the appendix), so that:

$$\hat{\rho}_{X_t} \otimes \hat{\rho}_{Y_t} \longrightarrow W \hat{\rho}_X W^\dagger$$
$$\mathrm{tr}_{\hat{\rho}_{Y_t}}\left(\hat{P}_y \hat{U}_2 W \hat{\rho}_{X_t} W^\dagger \hat{U}_2^\dagger \hat{P}_y^\dagger\right) \longrightarrow V_y \hat{P}_y \hat{U}_2 W \hat{\rho}_{X_t} W^\dagger \hat{U}_2^\dagger \hat{P}_y^\dagger V_y^\dagger$$

We can then construct Kraus operators such that $\hat{K}_y = V_y \hat{P}_y \hat{U}_2 W$. Figure 2b shows this updated circuit, where $\hat{U}_1$ is still the quantum implementation of the transition matrix and $\hat{K}_{y_t}$ is the quantum implementation of the Bayesian update after observation. This scheme to model a classical HMM can be written as:

$$\hat{\rho}_t = \frac{\hat{K}_{y_{t-1}}\left(\mathrm{tr}_{\hat{\rho}_{t-1}}(\hat{U}_1(\hat{\rho}_{t-1} \otimes \hat{\rho}_{X_t})\hat{U}_1^\dagger)\right)\hat{K}_{y_{t-1}}^\dagger}{\mathrm{tr}\left(\hat{K}_{y_{t-1}}\left(\mathrm{tr}_{\hat{\rho}_{t-1}}(\hat{U}_1(\hat{\rho}_{t-1} \otimes \hat{\rho}_{X_t})\hat{U}_1^\dagger)\right)\hat{K}_{y_{t-1}}^\dagger\right)} \tag{8}$$

We can similarly simplify $\hat{U}_1$ to a set of Kraus operators. We write the unitary operation $\hat{U}_1$ in terms of a set of n Kraus operators $\{\hat{K}_w\}$ as if we were to measure $\hat{\rho}_{t-1}$ immediately after the operation $\hat{U}_1$. However, instead of applying one Kraus operator associated with measurement as we do with Figure 2b, we sum over all of n possible 'observations', as if to 'ignore' the observation on $\hat{\rho}_{t-1}$. Post-multiplying each Kraus operator in $\{\hat{K}_w\}$ with each operator in $\{\hat{K}_y\}$, we have a set of Kraus operators $\{\hat{K}_{w_y,y}\}$ that can be used to model a classical HMM as follows (the full procedure is described in Algorithm 1):

$$\hat{\rho}_t = \frac{\sum_{w_y} \hat{K}_{w_y,y_{t-1}} \hat{\rho}_{t-1} \hat{K}_{w_y,y_{t-1}}^\dagger}{\mathrm{tr}\left(\sum_{w_y} \hat{K}_{w_y,y_{t-1}} \hat{\rho}_{t-1} \hat{K}_{w_y,y_{t-1}}^\dagger\right)} \tag{9}$$

We believe this procedure to be a useful illustration of performing classical operations on graphical models using quantum circuits. In practice, we needn't construct the Kraus operators in this peculiar fashion to simulate HMMs; an equivalent but simpler approach is to construct observable operators $\{\mathbf{T}_y\}$ from transition and

Algorithm 1 Simulating Hidden Markov Models with HQMMs

Input: Transition Matrix $\mathbf{A}$ and Emission Matrix $\mathbf{C}$
Output: Belief State as $diag(\hat{\rho})$, or $P(y_1, \ldots, y_n|\mathbb{D})$ where $\mathbb{D}$ is the HMM
Initialization:
  Let $s = \#$outputs, $n = \#$hidden states, $y_t =$ observed output at time t
  Prepare density matrix $\hat{\rho}_0$ in some initial state. $\hat{\rho}_0 = diag(\pi)$ if priors π are known.
  Construct unitary matrices $\hat{U}_1$ and $\hat{U}_2$ from $\mathbf{A}$ and $\mathbf{C}$ respectively using Algorithm 3 (in appendix)
  Using $\hat{U}_1$ and $\hat{U}_2$, construct a set of n Kraus Operators $\{\hat{K}_w\}$ and s Kraus operators $\{\hat{K}_y\}$, with $\hat{K}_w = V_w \hat{U}_1 W$ and $\hat{K}_y = V_y \hat{P}_y \hat{U}_2 W$ and combine them into a set $\{\hat{K}_{w_y,y}\}$ with $\hat{K}_{w_y,y} = \hat{K}_y \hat{K}_w$. (Matrix W tensors with an ancilla, Matrix V_y carries out a trivial partial trace operation and summing over V_w for all w carries out the proper partial trace operation. Details in appendix.)
for $t = 1 : T$ **do**
  $\hat{\rho}_{t+1} \leftarrow \sum_{w_y} \hat{K}_{w,y_t} \hat{\rho}_t (\hat{K}_{w_y,y_i})^\dagger$
end for
$\mathrm{tr}(\hat{\rho}_T)$ gives the probability of the sequence; renormalizing $\hat{\rho}_T$ gives the belief state on the diagonal.

emission matrices as described in section 2.3, and set the wth column of $\hat{K}^{(:,w)}_{w_y,y} = \sqrt{\mathbf{T}^{(:,w)}_y}$, with all other entries being zero. This ensures $\sum_{w_y,y} \hat{K}^\dagger_{w_y,y} K_{w_y,y} = \mathbb{I}$.

3.2 Formulating HQMMs

Monras et al. [2010] formulate Hidden Quantum Markov Models by defining a set of Kraus operators $\{\hat{K}_{w_y,y}\}$, where each observable y has w_y associated Kraus operators acting on a state with hidden dimension n, and they form a complete set such that $\sum_{w,y} \hat{K}^\dagger_{w,y} \hat{K}_{w,y} = \mathbb{I}$. The update rule for a quantum operation is exactly the same as Equation 9, which we arrived at by first constructing a quantum circuit to simulate HMMs with known parameters and then constructing operators $\{\hat{K}_{w,y}\}$ in a very peculiar way. The process outlined in the previous section is a particular parameterization of HQMMs to model HMMs. If we let the operators $\hat{U}_1$ and $\hat{U}_2$ be any unitary matrices, or the Kraus operators be any set of complex-valued matrices that satisfy $\sum_{w_y,y} \hat{K}^\dagger_{w_y,y} K_{w_y,y} = \mathbb{I}$, then we have a general and fully quantum HQMM.

Indeed, Equation 9 gives the forward algorithm for HQMMs. To find the probability of emitting an output y given the previous state $\hat{\rho}_{t-1}$, we simply take the trace of the numerator in Equation 9, i.e., $p(y_t|\hat{\rho}_{t-1}) = \mathrm{tr}\left(\sum_{w_y} \hat{K}_{w_y,y_t} \hat{\rho}_{t-1} \hat{K}^\dagger_{w_y,y_t}\right)$.

The number of parameters for a HQMM is determined by the number of latent states n, outputs s, and Kraus operators associated with an output w. To exactly simulate HMM dynamics with an HQMM, we need $w = n$ as per the derivation above. However, this constraint need not hold for a general HQMM, which can have any number of Kraus operators we apply and sum for a given output. w can also be thought of as the dimension of the ancilla $\hat{\rho}_{X_t}$ that we tensor with in Figure 2a before the unitary operation $\hat{U}_1$. Consequently, if we set $w = 1$, we do not tensor with an additional particle, but model the evolution of the original particle as unitary. In all, a HQMM requires learning n^2sw parameters, which is a factor w times more than a HMM with the observable operator representation which has n^2s parameters. The canonical representation of HMMs with with an $n \times n$ transition matrix and an $s \times n$ emission matrix has $n^2 + ns$ parameters.

HQMMs can also be seen as a complex-valued extension of norm-observable operator models defined by Zhao and Jaeger [2010]. Indeed, the HQMM we get by applying Algorithm 1 on a HMM is also a valid NOOM (allowing for multiple operators per output), implying that HMMs can be simulated by NOOMs. Both HMMs and NOOMs can be simulated by HQMMs (the latter is trivially true). While Zhao and Jaeger [2010] show that any NOOM can be written as an OOM, the exact relationship between HQMMs and OOMs requires further investigation.

4 AN ITERATIVE ALGORITHM FOR LEARNING HQMMs

We present an iterative maximum-likelihood algorithm to *learn* Kraus operators to model sequential data using an HQMM. Our algorithm is general enough that it can be applied to *any* quantum version of a classical machine learning algorithm for which the loss is defined in terms of the Kraus operators to be learned.

We begin by writing the likelihood of observing some sequence $y_1, \ldots, y_T$. Recall that for a given output y, we apply the w Kraus operators associated with that observable in the 'forward' algorithm, as $\sum_{w_y} \hat{K}_{w_y,y}(\cdot)\hat{K}_{w_y,y}$. If we do not renormalize the density matrix after applying these operators, the diagonal entries contain the joint probability of the corresponding system states and observing the associated sequence.

Algorithm 2 Iterative Learning Algorithm for Hidden Quantum Markov Models

Input: A $M \times \ell$ matrix Y, where $M = \#$ data points and $\ell =$length of a stochastic sequence to be modeled.

Output: A set of ws of $n \times n$ Kraus operators $\{\hat{K}_{w,s}\}$ that maximize the log-likelihood of the data, where n is the dimension of the hidden state, s is the number of outputs, and w is the number of operators per outputs.

1: **Initialization**: Randomly generate a set of ws Kraus operators $\{\hat{K}_{w,s}\}$ of dimension $n \times n$, and stack them vertically to obtain a matrix κ of dimension $nsw \times n$. Let b be the batch size, B the total number of batches to process, and Y_b a $b \times \ell$ matrix of randomly chosen data samples. Let $num\_iterations$ be the number of iterations spent modifying κ to maximize the likelihood of observing Y_b.

2: **for** batch $= 1{:}B$ **do**

3: Randomly select b sequences to process, and construct matrix Y_b

4: **for** $it = 1 : num\_iterations$ **do**

5: Randomly select rows i and j of κ to modify, $i < j$

6: Find $\vec{w} = (\phi, \psi, \delta, \theta)$ that maximises the log-likelihood of Y_b under the following update, and update:

$$\kappa^i \leftarrow \left(e^{i\phi/2} e^{i\psi} \cos(\theta)\right) \kappa^i + \left(e^{i\phi/2} e^{i\delta} \sin(\theta)\right) \kappa^j$$

$$\kappa^j \leftarrow \left(-e^{i\phi/2} e^{-i\delta} \sin(\theta)\right) \kappa^i + \left(e^{i\phi/2} e^{-i\psi} \cos(\theta)\right) \kappa^j$$

7: **end for**

8: **end for**

The trace of this un-normalized density matrix gives the probability of the sequence since we have summed over all the 'hidden' states. Thus, the log-likelihood of an HQMM predicting a sequence of length n is:

$$\mathcal{L} = \ln \operatorname{tr} \left(\sum_{w_{y_n}} \hat{K}_{w_{y_n}, y_n} \cdots \left(\sum_{w_{y_1}} \hat{K}_{w_{y_1}, y_1} \hat{\rho}_0 \hat{K}^\dagger_{w_{y_1}, y_1} \right) \cdots \hat{K}^\dagger_{w_{y_n}, y_n} \right) \tag{10}$$

It is not straightforward to directly maximize this log-likelihood using gradient descent; we must preserve the Kraus operator constraints and long sequences can quickly lead to underflow issues. Our approach is to learn a $nsw \times n$ matrix κ^*, which is essentially the set of ws Kraus operators $\{\hat{K}_{w,y}\}$ of dimension $n \times n$, stacked vertically. The Kraus operators constraint requires $\sum_s \hat{K}_s^\dagger \hat{K}_s = \mathbb{I}$, which implies $\kappa^\dagger \kappa = \mathbb{I}$, where the columns of κ are orthonormal.

Let κ be our guess and κ^* be the *true* matrix of stacked Kraus operators that maximizes the likelihood under the observed data. Then, there must exist some unitary operator $\hat{U}$ that maps κ to κ^*, i.e., $\kappa^* = \hat{U}\kappa$. Our goal is now to find the matrix $\hat{U}$. To do this, we use the fact that the matrix $\hat{U}$ can be written as the product of simpler matrices $\mathbf{H}(i, j, \theta, \phi, \psi, \delta)$ (proof in appendix):

$$\mathbf{H}(i,j,\theta,\phi,\psi,\delta) = \begin{bmatrix} 1 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & & \vdots & & \vdots \\ 0 & \cdots & e^{i\phi/2} e^{i\psi} \cos\theta & \cdots & e^{i\phi/2} e^{i\delta} \sin\theta & \cdots & 0 \\ \vdots & & \vdots & \ddots & \vdots & & \vdots \\ 0 & \cdots & -e^{i\phi/2} e^{-i\delta} \sin\theta & \cdots & e^{i\phi/2} e^{-i\psi} \cos\theta & \cdots & 0 \\ \vdots & & \vdots & & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & 0 & \cdots & 1 \end{bmatrix}$$

i and j specify the two rows in the matrix with the non-trivial entries, and the other paramters $\theta, \phi, \psi, \delta$ are angles that parameterize the non-trivial entries. The $\mathbf{H}$ matrices can be thought of as Givens rotations generalized for complex-valued unitary matrices. Applying such a matrix $\mathbf{H}(i, j, \theta, \phi, \psi, \delta)$ on κ has the effect of combining rows i and j $(i < j)$ of κ like so:

$$\kappa^i \leftarrow \left(e^{i\phi/2} e^{i\psi} \cos(\theta)\right) \kappa^i + \left(e^{i\phi/2} e^{i\delta} \sin(\theta)\right) \kappa^j$$

$$\kappa^j \leftarrow \left(-e^{i\phi/2} e^{-i\delta} \sin(\theta)\right) \kappa^i + \left(e^{i\phi/2} e^{-i\psi} \cos(\theta)\right) \kappa^j \tag{11}$$

Now the problem becomes one of identifying the sequence of $\mathbf{H}$ matrices that can take κ to κ^*. Since the optimization is non-convex and the $\mathbf{H}$ matrices need not commute, we are not guaranteed to find the global maximum. Instead, we look for a local-max κ^* that is reachable by only multiplying $\mathbf{H}$ matrices that increase the log-likelihood. To find this sequence, we iteratively find the parameters $(i, j, \theta, \phi, \psi, \delta)$ that, if used in Equation 11, would increase the log-likelihood. To perform this optimization, we use the `fmincon` function in MATLAB that uses interior-point optimization. It can also be computationally expensive to find the the best rows i, j to swap at a given step, so in our implementation, we randomly pick the rows (i, j) to swap. See Algorithm 2 for a summary. We believe more efficient implementations are possible, but we leave this to future work.

5 EXPERIMENTAL RESULTS

In this section, we evaluate the performance of our learning algorithm on simple synthetic datasets, and compare it to the performance of Expectation Maximization for HMMs (Rabiner [1989]). We judge the

quality of the learnt model using its Description Accuracy (DA) (Zhao and Jaeger [2007]), defined as:

$$DA = f\left(1 + \frac{\log_s P(Y|\mathbb{D})}{\ell}\right) \tag{12}$$

where ℓ is the length of the sequence, s is the number of output symbols in the sequence, Y is the data, and $\mathbb{D}$ is the model. Finally, the function $f(\cdot)$ is a nonlinear function that takes the argument from $(-\infty, 1]$ to $(-1, 1]$:

$$f(x) = \begin{cases} x & x \geq 0 \\ \frac{1-e^{-0.25x}}{1+e^{-0.25x}} & x < 0 \end{cases} \tag{13}$$

If $DA = 1$, the model perfectly predicted the stochastic sequence, while $DA > 0$ would mean that the model predicted the sequence better than random.

In each experiment, we generate 20 training sequences of length 3000, and 10 validation sequences of length 3000, with a 'burn-in' of 1000 to disregard the influence of the starting distribution. We use QETLAB (a MATLAB Toolbox developed by Johnston [2016]) to generate random HQMMs. We apply our learning algorithm once to learn HQMMs from data and report the DA. We use the Baum-Welch algorithm implemented in the `hmmtrain` function from MATLAB's Statistics and Machine Learning Toolbox to learn HMM parameters. When training HMMs, we train 10 models and report the best DA.

The first experiment compares learned models on data generated by a valid 'probability clock' NOOM/HQMM model (Zhao and Jaeger [2007]) that theoretically cannot be modeled by a finite-dimensional HMM. The second experiment considers a 2-state, 6-output HQMM which requires at least 4 classical states to model. These experiments are meant to showcase the greater expressiveness of HQMMs compared with HMMs, and we empirically demonstrate that our algorithm is able to learn an HQMM that can better predict the generated data than EM for classical HMMs.

5.1 Probability Clock

Zhao and Jaeger [2010] describes a 2-hidden state, 2-observable NOOM 'probability clock,' where the probability of generating an observable a changes periodically with the length of the sequence of as preceding it, and cannot be modeled with a finite-dimensional HMM:

$$\hat{K}_{1,1} = \begin{pmatrix} 0.6\cos(0.6) & -\sin(0.6) \\ 0.6\sin(0.6) & \cos(0.6) \end{pmatrix} \hat{K}_{1,2} = \begin{pmatrix} 0.8 & 0 \\ 0 & 0 \end{pmatrix} \tag{14}$$

This is a valid HQMM since $\sum_{y=1}^{y=2} K_{1,y}^\dagger K_{1,y} = \mathbb{I}$. Observe that this HQMM has only 1 Kraus operator per observable, which means it models the state evolution as unitary.

Our results in Table 2 demonstrate that a probability clock generates data that is hard for HMMs to model and that our iterative algorithm yields a simple HQMM that matches the predictive power of the original model.

Table 2: Performance of various HQMMs and HMMs learned from data generated by the probability clock model. HQMM parameters are given as (n, s, w) and HMM parameters are given as (n, s), where n is the number of hidden states, s is the number of observables, and w is the number of Kraus operators per observable. (T) indicates the true model, (L) indicates learned models. P is the number of parameters. Both the mean and STD of the DA are indicated for training and test data.

Model	P	Train DA	Test DA
$2,2,1-$HQMM (T)	8	0.1642 (0.0089)	0.1632 (0.0111)
$2,2,1-$HQMM (L)	8	0.1640 (0.0088)	0.1631 (0.0111)
$2,2-$HMM (L)	8	0.0851 (0.0074)	0.0833 (0.0131)
$4,2-$HMM (L)	24	0.1459 (0.0068)	0.1446 (0.0100)
$8,2-$HMM (L)	80	0.1639 (0.0087)	0.1630 (0.0108)

5.2 A Fully Quantum HQMM

Here, we present the results of our algorithm on a fully quantum HQMM. Since we use complex-valued entries, there is no known way of writing our model as an equivalent-sized HMM or observable operator model.

We motivate this model with a physical system. Consider electron spin: quantized angular momentum that can either be 'up' or 'down' along whichever spatial axis the measurement is made, but not in between. There is no well-defined 3D vector describing electron spin along the 3 spatial dimensions, only 'up' or 'down' along a chosen axis of measurement (i.e., measurement basis). This is unlike classical angular momentum which can be represented by a vector with well-defined components in three spatial dimensions. Picking an arbitrary direction as the z-axis, we can write the electron's spin state in the $\{+\mathbf{z}, -\mathbf{z}\}$ basis so that $\begin{bmatrix}1 & 0\end{bmatrix}^T$ is $|+\mathbf{z}\rangle$ and $\begin{bmatrix}0 & 1\end{bmatrix}^T$ is $|-\mathbf{z}\rangle$. But electron spin constitutes a two-state quantum system, so it can be in superpositions of the orthogonal 'up' and 'down' quantum states, which can be parameterized with (θ, ϕ) and written as $|\psi\rangle = \cos\left(\frac{\theta}{2}\right)|+\mathbf{z}\rangle + e^{i\phi}\sin\left(\frac{\theta}{2}\right)|-\mathbf{z}\rangle$, where $0 \leq \theta \leq \pi$ and $0 \leq \phi \leq 2\pi$. The Bloch sphere (sphere with radius 1) is a useful tool to visualize qubits since it can map any two-state system to a point on the surface of the sphere using (θ, ϕ) as polar and azimuthal angles. We could also have chosen $\{+\mathbf{x}, -\mathbf{x}\}$ or $\{+\mathbf{y}, -\mathbf{y}\}$, which can be written in our original basis:

$$|+\mathbf{x}\rangle = \frac{1}{\sqrt{2}}|+\mathbf{z}\rangle + \frac{1}{\sqrt{2}}|-\mathbf{z}\rangle \qquad \left(\theta = \frac{\pi}{2}, \phi = 0\right) \tag{15}$$

$$|-\mathbf{x}\rangle = \frac{1}{\sqrt{2}}|+\mathbf{z}\rangle - \frac{1}{\sqrt{2}}|-\mathbf{z}\rangle \qquad \left(\theta = \frac{\pi}{2}, \phi = \pi\right) \tag{16}$$

$$|+\mathbf{y}\rangle = \frac{1}{\sqrt{2}}|+\mathbf{z}\rangle + \frac{i}{\sqrt{2}}|-\mathbf{z}\rangle \qquad \left(\theta = \frac{\pi}{2}, \phi = \frac{\pi}{2}\right) \tag{17}$$

$$|-\mathbf{y}\rangle = \frac{1}{\sqrt{2}}|+\mathbf{z}\rangle - \frac{i}{\sqrt{2}}|-\mathbf{z}\rangle \qquad \left(\theta = \frac{\pi}{2}, \phi = \frac{3\pi}{2}\right) \tag{18}$$

Now consider the following process, inspired by the Stern-Gerlach experiment (Gerlach and Stern [1922]) from quantum mechanics. We begin with an electron whose spin we represent in the $\{+\mathbf{z}, -\mathbf{z}\}$ basis. At each time step, we pick one of the x, y, or z directions uniformly and at random, and apply an inhomogeneous magnetic field along that axis. This is an act of measurement that collapses the electron spin to either 'up' or 'down' along that axis, which will deflect the electron in that direction. Let us use the following encoding scheme for the results of the measurement: 1: $+\mathbf{z}$, 2: $-\mathbf{z}$, 3: $+\mathbf{x}$, 4: $-\mathbf{x}$, 5: $+\mathbf{y}$, 6: $-\mathbf{y}$. Consequently, at each time step, the observation tells us which axis we measured along, and whether the spin of the particle is now 'up' or 'down' along that axis. As an example, if we prepare an electron spin 'up' along the z-axis, and observe the following sequence: $1, 3, 2, 6$, it means that we applied the inhomogeneous magnetic field in the z-direction, then x-direction, then z-direction, and finally the y-direction, causing the electron spin state to evolve as $+\mathbf{z}, +\mathbf{x}, -\mathbf{z}, -\mathbf{y}$. Note that transitions $1 \leftrightarrow 2$, $3 \leftrightarrow 4$, and $5 \leftrightarrow 6$ are not allowed, since there are no spin-flip operations in our process. Admittedly, this is a slightly contrived example, since normally we think of a hidden state that evolves according to some rules, producing noisy observation. Here, we select the observation (down to the pair, $(1,2)$, $(3,4)$, $(5,6)$) that we wish to observe, and that tells us how the 'hidden state' evolves as described by a chosen basis.

This model is related to the 2-state HQMM requiring 3 classical states described in Monras et al. [2010]. It is still a 2-state system, but we add two new Kraus operators with complex entries and renormalize:

$$\hat{K}_{1,1} = \begin{pmatrix} \frac{1}{\sqrt{3}} & 0 \\ 0 & 0 \end{pmatrix} \qquad \hat{K}_{1,2} = \begin{pmatrix} 0 & 0 \\ 0 & \frac{1}{\sqrt{3}} \end{pmatrix} \tag{19}$$

$$\hat{K}_{1,3} = \begin{pmatrix} \frac{1}{2\sqrt{3}} & \frac{1}{2\sqrt{3}} \\ \frac{1}{2\sqrt{3}} & \frac{1}{2\sqrt{3}} \end{pmatrix} \qquad \hat{K}_{1,4} = \begin{pmatrix} \frac{1}{2\sqrt{3}} & -\frac{1}{2\sqrt{3}} \\ -\frac{1}{2\sqrt{3}} & \frac{1}{2\sqrt{3}} \end{pmatrix} \tag{20}$$

$$\hat{K}_{1,5} = \begin{pmatrix} \frac{1}{2\sqrt{3}} & -\frac{i}{2\sqrt{3}} \\ \frac{i}{2\sqrt{3}} & \frac{1}{2\sqrt{3}} \end{pmatrix} \qquad \hat{K}_{1,6} = \begin{pmatrix} \frac{1}{2\sqrt{3}} & \frac{i}{2\sqrt{3}} \\ -\frac{i}{2\sqrt{3}} & \frac{1}{2\sqrt{3}} \end{pmatrix} \tag{21}$$

Physically, Kraus operators $\hat{K}_{1,1}$ and $\hat{K}_{1,2}$ keep the spin along the z-axis, Kraus operators $\hat{K}_{1,3}$ and $\hat{K}_{1,4}$ rotate the spin to lie along the x-axis, while Kraus operators $\hat{K}_{1,5}$ and $\hat{K}_{1,6}$ rotate the spin to lie along the y-axis. Following the approach of Monras et al. [2010], we write down an equivalent 6-state HMM, and compute the rank of a Hankel matrix with the statistics of this process, yielding a requirement of 4 classical states as a weak lower bound.

We present the results of our learning algorithm applied to data generated by this model in Table 3. We find that our algorithm can learn a 2-state HQMM (same size as the model that generated the data) with predictive power matched only by a 6-state HMM.

Table 3: Performance of various HQMMs and HMMs on the fully quantum HQMM. HQMM parameters are given as (n, s, w) and HMM parameters are given as (n, s), where n is the number of hidden states, s is the number of observables, and w is the number of Kraus operators per observable

Model	P	Train DA	Test DA
$2, 6, 1-$HMM (T)	24	0.1303 (0.0042)	0.1303 (0.0047)
$2, 6, 1-$HQMM (L)	24	0.1303 (0.0042)	0.1301 (0.0047)
$2, 6-$HMM (L)	16	0.0327 (0.0038)	0.0328 (0.0033)
$3, 6-$HMM (L)	27	0.0522 (0.0043)	0.0530 (0.0040)
$4, 6-$HMM (L)	40	0.0812 (0.0042)	0.0822 (0.0045)
$5, 6-$HMM (L)	55	0.0967 (0.0042)	0.0967 (0.0045)
$6, 6-$HMM (L)	72	0.1305 (0.0042)	0.1301 (0.0049)

5.3 Discussion

Interestingly, we are able to learn reasonable models with $w = 1$, i.e., modeling state evolution as unitary. Indeed, the probability clock and Stern-Gerlach inspired model assume unitary state evolution, and these HQMMs can model the same sequence with far fewer parameters compared to an HMM. We provide additional experimental results in the appendix that show that we can learn the 2-state HQMM presented by Monras et al. [2010] from data. In our experiments on HMM-generated data, we found that small HQMMs outperform HMMs with the same number of hidden states, although the parameter count ends up being larger (see appendix for results). As model size increases, our HQMMs are over-parameterized, becoming prone to getting stuck in local optima, and EM for HMMs may work better in practice on HMM-generated data.

6 CONCLUSION

We formulated and parameterized HQMMs by first finding quantum circuits to implement HMMs and relaxing some constraints. We showed how quantum analogues of classical conditioning and marginalization can be implemented, and these methods are general enough to construct quantum versions of any probabilistic graphical model. We also proposed an iterative maximum-likelihood algorithm to learn the Kraus operators for HQMMs. We demonstrated that our algorithm could successfully learn HQMMs that were shown to (theoretically) better model certain sequences in the literature. While our HQMMs cannot model data any better than a sufficiently large HMM, we find that HQMMs can better model the same data with fewer hidden states. Future work could look at optimizing our algorithm to scale on larger datasets, and determining more generally when HQMMs are more suitable than HMMs. We speculate that quantum models could lead to improvements in areas where 'quantum' effects may better model the dynamic processes.

Acknowledgements

We thank Theresa W. Lynn for her advice and input on this work.

References

Jacob Biamonte, Peter Wittek, Nicola Pancotti, Patrick Rebentrost, Nathan Wiebe, and Seth Lloyd. Quantum machine learning. *arXiv preprint arXiv:1611.09347*, 2016.

Lewis A Clark, Wei Huang, Thomas M Barlow, and Almut Beige. Hidden quantum markov models and open quantum systems with instantaneous feedback. In *ISCS 2014: Interdisciplinary Symposium on Complex Systems*, pages 143–151. Springer, 2015.

Walther Gerlach and Otto Stern. Der experimentelle nachweis der richtungsquantelung im magnetfeld. *Zeitschrift für Physik*, 9(1):349–352, 1922.

Herbert Jaeger. Observable operator models for discrete stochastic time series. *Neural Computation*, 12(6):1371–1398, 2000.

Nathaniel Johnston. QETLAB: A MATLAB toolbox for quantum entanglement, version 0.9. `http://qetlab.com`, January 2016.

Alex Monras, Almut Beige, and Karoline Wiesner. Hidden quantum markov models and non-adaptive read-out of many-body states. *arXiv preprint arXiv:1002.2337*, 2010.

Lawrence R Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

Maria Schuld, Ilya Sinayskiy, and Francesco Petruccione. An introduction to quantum machine learning. *Contemporary Physics*, 56(2):172–185, 2015a.

Maria Schuld, Ilya Sinayskiy, and Francesco Petruccione. Simulating a perceptron on a quantum computer. *Physics Letters A*, 379(7):660–663, 2015b.

Nathan Wiebe, Ashish Kapoor, and Krysta Svore. Quantum perceptron models. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 3999–4007. Curran Associates, Inc., 2016. URL `http://papers.nips.cc/paper/6401-quantum-perceptron-models.pdf`.

Ming-Jie Zhao and Herbert Jaeger. Norm observable operator models. Technical report, Jacobs University, 2007.

Ming-Jie Zhao and Herbert Jaeger. Norm-observable operator models. *Neural computation*, 22(7):1927–1959, 2010.