KOOPMAN OPERATOR LEARNING FOR ACCELERATING QUANTUM OPTIMIZATION AND MACHINE LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Finding efficient optimization methods plays an important role for quantum optimization and quantum machine learning on near-term quantum computers. While backpropagation on classical computers is computationally efficient, obtaining gradients on quantum computers is not, because the computational complexity scales linearly with the number of parameters and measurements. In this paper, we connect Koopman operator theory, which has been successful in predicting nonlinear dynamics, with natural gradient methods in quantum optimization. We propose a data-driven approach using Koopman operator learning to accelerate quantum optimization and quantum machine learning. We develop two new families of methods: the sliding window dynamic mode decomposition (DMD) and the neural DMD for efficiently updating parameters on quantum computers. We show that our methods can predict gradient dynamics on quantum computers and accelerate the quantum variational eigensolver used in quantum optimization, as well as quantum machine learning. We further implement our Koopman operator learning algorithm on a real quantum computer and demonstrate their practical effectiveness.

1 INTRODUCTION

There has been rapid development of quantum technologies and quantum computation in recent years. A number of efforts have been put into demonstrating quantum advantages and speedup. Quantum optimization (Moll et al., 2018) and quantum machine learning (QML) (Biamonte et al., 2017), as important applications of quantum technologies, have received increased interest. The Variational Quantum Eigensolver (VQE) (Peruzzo et al., 2014b; Tilly et al., 2022), as a quantum optimization algorithm, has been developed and applied to understanding problems in high energy physics (Klco et al., 2018; Rinaldi et al., 2022), condensed matter physics (Wecker et al., 2015), and quantum chemistry (Peruzzo et al., 2014a). The Variational Quantum Algorithm (VQA) (Cerezo et al., 2021) such as the Quantum Approximate Optimization Algorithm (QAOA) (Farhi et al., 2014) has been applied to the max-cut problem. A recent experiment on 289 qubits has demonstrated a powerful VQA application in classical optimization by benchmarking against a variety of classical algorithms (Ebadi et al., 2022). Quantum machine learning (QML) has also been developed for various tasks including supervised learning (Havlíček et al., 2019), unsupervised learning (Kerenidis et al., 2019) and reinforcement learning (Dong et al., 2008). Theoretical advantages of quantum machine learning have been investigated (Huang et al., 2022b; Liu et al., 2021), and experiments on real quantum computers have demonstrated encouraging progress (Huang et al., 2022a; Rudolph et al., 2022).

In the noisy intermediate-scale quantum (NISQ) era (Preskill, 2018), due to the noisy nature of current quantum computer architectures, hybrid classical-quantum schemes have been proposed for quantum optimization and quantum machine learning and become a prominent approach. The key spirit of the hybrid approach relies on performing optimization and machine learning on parameterized quantum circuits with quantum features while updating the parameters in the circuit is done through classical computers. In classical machine learning, backprogation only requires vector Jacobian calculations which share the same complexity as the forward evaluation. Obtaining gradients under the hybrid scheme is much more challenging. Calculating gradients in quantum computers is challenging for two reasons: (1) gradient calculation typically scales linearly in the number of

parameters; and (2) the quantum nature of the gradient itself entails sampling over repeated measurements. Despite various research on quantum optimization and quantum machine learning in simulation, the implementation of gradient-based methods on real quantum computers is computationally inefficient which limits their applications in practice. It is an important open problem in the field to develop scalable and efficient optimization methods for quantum optimization applications and quantum machine learning tasks.

In this work, we propose Koopman operator learning for accelerating quantum optimization and quantum machine learning. The Koopman operator theory is a powerful framework for understanding and predicting nonlinear dynamics through linear dynamics embedded into a higher dimensional space (Brunton et al., 2021). By viewing parameter optimization on quantum computers as a nonlinear dynamical evolution in the parameter space, we connect gradient dynamics in quantum optimization to the Koopman operator theory. In particular, the quantum natural gradient method has a natural embedding through the quantum computer parameterization into a Hilbert space, which is related to the linear imaginary time evolution of the quantum system.

With the guidance of Koopman operator theory, we develop new Koopman operator learning algorithms for quantum optimization and quantum machine learning, including the sliding window dynamic mode decomposition (SW-DMD) and neural-network-based DMD that learns the Koopman embedding via a neural network parameterization. Our approach is data-driven and based on the information of only a few gradient steps such that the cost of prediction does not scale with the number of parameters.

Our methods enable efficient learning of gradient dynamics for accelerating quantum optimization and quantum machine learning. Our experiments are both on numerical simulations and real quantum computers. We first demonstrate our Koopman operator learning algorithms for the quantum variational eigensolver, VQE, which is an important application in quantum optimization. We test the methods for the natural gradient and Adam (Kingma & Ba, 2014) optimizers on quantum Ising model simulations and demonstrate their success on a real quantum computer provided by IBM with a quasi-gradient-based optimizer Simultaneous Perturbation Stochastic Approximation (SPSA) (Spall, 1992). Finally, we apply our methods to accelerate hybrid quantum machine learning on the MNIST dataset.

2 Related Work

Koopman operators. Koopman operator theory (Koopman, 1931) was first proposed by Koopman and von Neumann in early 1930s to understand dynamical systems. Dynamic mode decomposition (DMD) (Schmid, 2010) was developed to learn the Koopman operator under the linear dynamics assumption of the observed data. Later, more advanced methods such as the extended-DMD based on kernel methods (Baddoo et al., 2022) and dictionary learning (Li et al., 2017) were introduced to go beyond the linear dynamics assumption, and achieved better performance. Recently, machine learning methods were integrated into Koopman operator learning where neural networks are used to learn the mapping to high dimensional space, in which the dynamics becomes linear (Lusch et al., 2018; Li et al., 2019). The machine learning Koopman operator methods were shown to learn nonlinear differential equation dynamics successfully.

In addition to predicting nonlinear dynamics, recently Koopman operator theory was applied to optimize neural network training (Dogra & Redman, 2020; Tano et al., 2020) and pruning (Redman et al., 2021). These works take the perspective of viewing the optimization process of neural networks as a nonlinear dynamical evolution and uses dynamic mode decomposition to predict the parameter updates in the future. In a more empirical study, Sinha et al. (2017) trained a convolutional neural network (CNN) to predict the future weights of neural networks, trained on standard vision tasks.

Besides important applications for classical systems, the Koopman operator theory has natural connections to quantum mechanics. Recently, researchers considered Koopman operator theory for quantum control (Goldschmidt et al., 2021) and prediction of one particle quantum system evolution (Klus et al., 2022). Since quantum mechanical systems provide a natural high dimensional Hilbert space through the wave function, the theory was considered for embedding classical equations for learning and solving differential equations (Lin et al., 2022; Giannakis et al., 2022).

Optimization methods on quantum computers. Due to the nature of hybrid classical-quantum algorithms, taking gradients on quantum computers cannot be done as efficiently as backpropagation on classical computers. In general, the complexity of obtaining gradients scales with the number of parameters on the quantum computer and the number of measurements per parameter. To perform optimization on quantum computers, gradient-free methods such as SPSA and COBYLA (Powell, 1994) are used though they may not scale well to quantum circuits with large number of parameters. For gradient methods, Adam is a common choice for better scaling though its complexity scales with the number of parameters as discussed above. There are also higher order methods such as the quantum natural gradient method which have been shown to have faster convergence while being more challenging to realize experimentally.

Our work. We contribute to the development of an efficient data-driven approach that addresses the problems above via a Koopman operator learning approach for accelerating optimization on quantum computers. We integrate the state-of-the-art machine learning Koopman operator with the insight of quantum dynamics to achieve our goal. There are several important features of our work that distinguish it from the relevant works above. First, our goal is to accelerate quantum optimization and quantum machine learning, which compared to classical neural networks has much higher complexity of taking gradients, such that the gain of acceleration will be much more prominent. Second, instead of predicting the gradient dynamics directly, we focus on decreasing the loss function during the training and develop iterative optimal prediction algorithms, which is much more flexible and effective than the one-time prediction in the previous work (Dogra & Redman, 2020). Third, the previous literature on Koopman operators for neural networks applies only the standard DMD which may not be well situated for complicated nonlinear training dynamics. Instead, we develop and investigate the sliding window DMD (SW-DMD) and several variants of neural DMD, including the multi-layer perceptron DMD (MLP-DMD), MLP-SW-DMD and CNN-DMD. We show that our new algorithms have much better performance compared to the standard DMD, which also provides potential for accelerating optimization in general, and enables novel avenues of research, previously unexplored by the related works.

3 Connecting Koopman Operator Theory and Quantum Optimization

3.1 Koopman Operator Theory

We consider a dynamical system with a collection of state variables $\{x(t) \in \mathbb{R}^n\}$ with a transition function T such that $x(t+1) = T(x(t))$. The Koopman operator theory developed by Koopman asserts that there exists a linear operator $\mathcal{K}$ and a function g such that

$$\mathcal{K}g(x(t)) = g(T(x(t))) = g(x(t+1)) \tag{1}$$

where $\mathcal{K}$ is the Koopman operator. In general, the Koopman operator can act on an infinite-dimensional space. When the Koopman operator is constrained to a finite dimensional invariant subspace with $g : \mathbb{R}^n \to \mathbb{R}^m$, the Koopman operator $\mathcal{K}$ can be presented as a Koopman matrix $K \in \mathbb{R}^{m \times m}$.

We now can search for the function g. The standard DMD takes g to be the identity function with the assumption that the underlying dynamics of x is approximately linear, *i.e.*, T is a linear operator. The extended-DMD method utilizes other feature functions such as polynomial and trigonometric functions as the basis functions for g. Recently, machine learning methods for the Koopman operator adopt neural networks as universal approximators for learning g and achieve better performance (Lusch et al., 2018).

3.2 Variational Quantum Eigensolver (VQE)

Consider a Hamiltonian $\mathcal{H}$ describing interactions in a physical system. For a quantum mechanical system with N spins or qubits, $\mathcal{H}$ is a Hermitian operator acting on the 2^N-dimesional Hilbert space for wave functions. A wave function ψ is an l_2-normalized complex-valued vector that contains all the information of the state of the system. In particular, the energy of the system is given by a loss function $\mathcal{L}(\psi) = \langle \psi, \mathcal{H}\psi \rangle$. VQE encodes the wave function as $\psi_{\boldsymbol{\theta}}$ by a set of parameters

$\boldsymbol{\theta} \in \mathbb{R}^{n_{\text{params}}}$ via an ansatz layer on a quantum circuit in the quantum computer, as is shown in the top left part of Figure 1. n_{params} is usually chosen to be polynomial in the number of qubits N, which scales much slower than the 2^N-scaling of the dimension of ψ itself in the original Hilbert space. The goal of VQE is to minimize the loss by minimizing for $\boldsymbol{\theta}$ in the following objective

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}) = \arg\min_{\boldsymbol{\theta}} \langle \psi_{\boldsymbol{\theta}}, \mathcal{H}\psi_{\boldsymbol{\theta}} \rangle. \tag{2}$$

Since the dimension of $\boldsymbol{\theta}$ is usually smaller than the dimension of ψ itself, the loss $\mathcal{L}(\boldsymbol{\theta}^*)$ may still differ from the minimum of $\mathcal{L}(\psi)$. However, thanks to the nonlinearity in the encoding $\boldsymbol{\psi_\theta}$ from the nonlinear ansatz layer, in many cases $\boldsymbol{\theta}^*$ can still be a very good approximation of the true minimum of $\mathcal{L}(\psi)$. Quantum machine learning has a similar setup that targets at minimizing $\mathcal{L}(\boldsymbol{\theta})$, and parameterized quantum circuits in QML are called quantum neural networks.

We may use a gradient-based classical optimizer such as Adam to minimize the loss function, and this requires attaining the gradient $\frac{\partial \mathcal{L}}{\partial \theta_i}$. In classical machine learning, the computational cost of backpropagation is only $O(1)$ per iteration. However, in the quantum case, we have to explicitly evaluate the loss with a perturbation in each direction i, for example by using the parameter-shift rule (Mitarai et al., 2018; Schuld et al., 2019) $(\mathcal{L}(\theta_i + \pi/2) - \mathcal{L}(\theta_i - \pi/2))/2$, which leads to an $O(n_{\text{params}})$ computational cost per iteration. Hence, quantum optimization is significantly more expensive than classical optimization. As another note, the classical computational components involved in VQE, even including training neural-network-based algorithms in the following sections, typically are much cheaper than the quantum cost.

3.3 Quantum Fisher Information and Quantum Natural Gradient

The quantum natural gradient method (Stokes et al., 2020) is a generalization of the classical natural gradient method in classical machine learning (Amari, 1998), where the probability is generalized to the complex-valued wave function. The natural gradient for parameter θ update for Eq. 2 is given by a nonlinear differential equation

$$\frac{d}{dt}\boldsymbol{\theta}(t) = -\eta F^{-1} \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}(t)), \tag{3}$$

where η is the scalar learning rate, and F is the quantum Fisher Information matrix given by $F_{ij} = \langle \frac{\partial \psi_{\boldsymbol{\theta}}}{\partial \theta_i}, \frac{\partial \psi_{\boldsymbol{\theta}}}{\partial \theta_j} \rangle - \langle \frac{\partial \psi_{\boldsymbol{\theta}}}{\partial \theta_i}, \psi_{\boldsymbol{\theta}} \rangle \langle \psi_{\boldsymbol{\theta}}, \frac{\partial \psi_{\boldsymbol{\theta}}}{\partial \theta_j} \rangle$. It can be shown that the above nonlinear differential equation for $\boldsymbol{\theta}$ is equivalent to the dynamical equation of $\psi_{\boldsymbol{\theta}}(t)$ as follows

$$\frac{d\psi_{\boldsymbol{\theta}}(t)}{dt} = -\mathbb{P}_{\psi_{\boldsymbol{\theta}}} \mathcal{H} \psi_{\boldsymbol{\theta}}(t), \tag{4}$$

where $\mathbb{P}_{\psi_\theta}$ is a projection operator onto the manifold given by the parameterized quantum circuit (Hackl et al., 2020).

Notice that $\mathcal{H}$ is a linear operator. When the parametrized quantum circuit is sufficiently expressive such that the projection is within the manifold, Eq. 4 can be approximated by a linear differential equation. By viewing the parameters $\boldsymbol{\theta}(t)$ as the state variable $x(t)$ in the Koopman theory, the quantum circuit naturally generates a (wave) function $\psi_{\boldsymbol{\theta}}$ for $\boldsymbol{\theta}$ that playes the role of g in Eq. 1, whose dynamics is close to linear in the Hilbert space. The dimension of $\psi_{\boldsymbol{\theta}}$ is 2^N, and we do not construct it explicitly. However, the existence of $\psi_{\boldsymbol{\theta}}$ and its approximate linear dynamics builds the theoretical foundation for Koopman operator learning algorithms for $\boldsymbol{\theta}(t)$ in optimization and the application of accelerating the parameter updates.

4 Koopman Operator Learning Algorithms

4.1 Our framework: data-driven approach for accelerating optimization

Our algorithmic scheme of Koopman learning for quantum optimization and quantum machine learning is shown in Figure 1. In quantum optimization and quantum machine learning, a quantum computer uses quantum measurements to evaluate the loss function $\mathcal{L}$ at the initial parameter $\boldsymbol{\theta}(t_0)$, which is an input into the ansatz gate. Then, we use an optimizer on a classical computer,

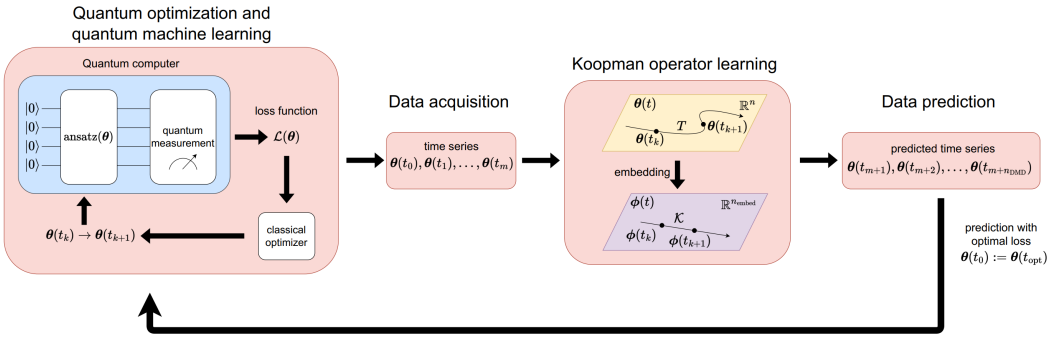


Figure 1: Hybrid quantum-classical architecture scheme of Koopman learning for quantum optimization and quantum machine learning.

such as Adam to update the parameter $\boldsymbol{\theta}$, thus incrementing the optimization iteration by 1. If the optimizer is gradient-based, for each iteration, $O(n_{\text{params}})$ additional quantum measurements are needed to evaluate the gradient. Next, we input the updated parameters into the ansatz and repeat this procedure for m steps, and thus acquire a time sequence $\boldsymbol{\theta}(t_0), \boldsymbol{\theta}(t_1), ..., \boldsymbol{\theta}(t_m)$. We then the apply Koopman operator learning algorithms to the acquired data and predict the time series for n_{DMD} steps to obtain $\boldsymbol{\theta}(t_{m+1}), \boldsymbol{\theta}(t_{m+2}), ..., \boldsymbol{\theta}(t_{m+n_{\text{DMD}}})$. The loss at these predicted parameters can also be evaluated on the quantum computer, but no gradient measurement is needed. From these n_{DMD} loss function values, we find the lowest loss and the corresponding time t_{opt}. The optimal Koopman-predicted parameter $\boldsymbol{\theta}(t_{\text{opt}})$ is then used as the initial point for the next quantum optimization and quantum machine learning. This completes a full cycle of our algorithm, and we can repeat the cycle until the loss reaches our target.

4.2 Sliding Window DMD (SW-DMD)

The standard DMD (Brunton et al. (2021)) uses a linear fit for the dynamics in the original space for the column vector $\boldsymbol{\theta} \in \mathbb{R}^n$

$$\boldsymbol{\theta}(t_{k+1}) = K\boldsymbol{\theta}(t_k). \tag{5}$$

We can concatenate $\boldsymbol{\theta}$ at successive times to obtain two data matrices

$$\boldsymbol{\Theta}(t_0) = [\boldsymbol{\theta}(t_0) \quad \boldsymbol{\theta}(t_1) \quad \cdots \quad \boldsymbol{\theta}(t_m)], \quad \boldsymbol{\Theta}(t_1) = [\boldsymbol{\theta}(t_1) \quad \boldsymbol{\theta}(t_2) \quad \cdots \quad \boldsymbol{\theta}(t_{m+1})], \tag{6}$$

where $\boldsymbol{\Theta}(t_1)$ is the one-step time evolution of $\boldsymbol{\Theta}(t_0)$. In the case of approximate linear dynamics, the matrix K is the same for all times t_k, and then Eq. 5 becomes

$$\boldsymbol{\Theta}(t_1) \approx K\boldsymbol{\Theta}(t_0), \tag{7}$$

where $K \in \mathbb{R}^{n\times n}$. The best fit is at the minimum of the Frobenius loss

$$K = \arg\min_K \|\boldsymbol{\Theta}(t_1) - K\boldsymbol{\Theta}(t_0)\|_F = \boldsymbol{\Theta}(t_1)\boldsymbol{\Theta}(t_0)^+, \tag{8}$$

where $^+$ is the pseudo-inverse.

When the dynamics of $\boldsymbol{\theta}$ is not linear, we can instead consider a time-delayed embedding with a sliding window and concatenate the steps to form an extended data matrix (Dylewsky et al., 2022)

$$\boldsymbol{\Phi}(\boldsymbol{\Theta}(t_0)) = [\boldsymbol{\phi}(t_0) \quad \boldsymbol{\phi}(t_1) \quad \cdots \quad \boldsymbol{\phi}(t_m)] = \begin{bmatrix} \boldsymbol{\theta}(t_0) & \boldsymbol{\theta}(t_1) & \cdots & \boldsymbol{\theta}(t_m) \\ \boldsymbol{\theta}(t_1) & \boldsymbol{\theta}(t_2) & \cdots & \boldsymbol{\theta}(t_{m+1}) \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{\theta}(t_d) & \boldsymbol{\theta}(t_{d+1}) & \cdots & \boldsymbol{\theta}(t_{m+d}) \end{bmatrix}. \tag{9}$$

$\boldsymbol{\Phi}$ is generated by a sliding window of size $d+1$ at $m+1$ consecutive time steps. Each column of $\boldsymbol{\Phi}$ is a time-delayed embedding for $\boldsymbol{\Theta}$, and the different columns $\boldsymbol{\phi}$ in $\boldsymbol{\Phi}$ are embeddings at different starting times. The time-delay embedding captures some nonlinearity in the dynamics of $\boldsymbol{\theta}$, with

$$\boldsymbol{\Theta}(t_{d+1}) \approx K\boldsymbol{\Phi}(\boldsymbol{\Theta}(t_0)), \tag{10}$$

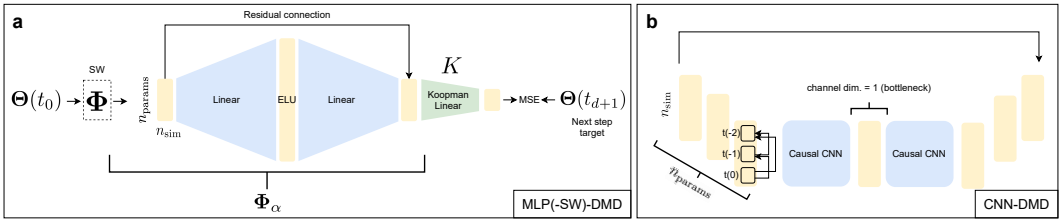


Figure 2: Neural network architectures for our neural DMD approaches. (a) MLP bottleneck architecture with MSE loss for training. (b) CNN bottleneck architecture that operates on simulations as temporal dimension and parameters as channel dimension.

where $K \in \mathbb{R}^{n \times n(d+1)}$. The best fit is given by

$$K = \arg\min_K \|\mathbf{\Theta}(t_{d+1}) - K\mathbf{\Phi}(\mathbf{\Theta}(t_0))\|_F = \mathbf{\Theta}(t_{d+1})\mathbf{\Phi}(\mathbf{\Theta}(t_0))^+. \tag{11}$$

The used data from the acquired time series with the largest time in the above equation is $\boldsymbol{\theta}(t_{m+d+1})$. When making prediction, we start with $\boldsymbol{\theta}(t_{m+d+2}) = K\boldsymbol{\phi}(t_{m+1})$. Then we update from $\boldsymbol{\phi}(t_{m+1})$ to $\boldsymbol{\phi}(t_{m+2})$ by removing the oldest data $\boldsymbol{\theta}(t_{m+1})$ and adding the newly predicted data $\boldsymbol{\theta}(t_{m+d+2})$. We repeat this prediction using $\boldsymbol{\theta}(t_{k+d+1}) = K\boldsymbol{\phi}(t_k)$ iteratively. Our approach is different from the approach used by Dylewsky et al. (2022), as we do not perform an additional singular value decomposition before doing DMD and our matrix K is non-square.

We denote DMD performed this way as sliding window DMD (SW-DMD). The standard DMD is a special case of SW-DMD when the sliding window size is 1 (*i.e.*, $d = 0$).

4.3 Neural DMD

General formulation. In this section we will describe our study on how to improve DMD with a neural approach, and potentially provide a better approximation to the nonlinear dynamics. We ask whether the hard-coded sliding window transformation $\mathbf{\Phi}$ can be a neural network. Thus, by simply reformulating $\mathbf{\Phi}$ in Eq. 11 as a neural network, we formulate a natural neural objective for Koopman operator learning as follows

$$\arg\min_{K,\alpha} \|\mathbf{\Theta}(t_{d+1}) - K\mathbf{\Phi}_\alpha(\mathbf{\Theta}(t_0))\|_F, \tag{12}$$

where $K \in \mathbb{R}^{N_{in} \times N_{out}}$ is a linear Koopman operator and $\mathbf{\Phi}_\alpha(\mathbf{\Theta}(t_0))$ is a nonlinear neural embedding by a neural network $\mathbf{\Phi}_\alpha$ with parameters α. More specifically, $\mathbf{\Phi}_\alpha := \mathrm{NN}_\alpha \circ \mathbf{\Phi}$ is a composition of the neural network architecture NN_α and the sliding window embedding $\mathbf{\Phi}$ from the previous section. The parameters K and α are trained jointly on Eq. 12 from scratch with every new batch of optimization history by using the Adam optimizer for 30k steps with 9k steps of linear warmup from 0 to 0.001 and then cosine decay back to 0 at step 30k. We use the MSE loss, which minimizes the Frobenius norm.

Choice of architecture for $\mathbf{\Phi}$. Our study will guide us in constructing a family of neural network architectures for neural DMD, which we list below (see also Appendix B for more details).

MLP-DMD. We first begin exploring a simple MLP architecture, as shown in Figure 2 (a). The result is MLP-DMD. The architecture consists of two linear layers with an ELU (Clevert et al., 2015) activation on the hidden layer and a residual connection. We use an expansion ratio for the hidden layer and explored ratios in $\{1,2\}$, but selected 1 as the best one, possibly because higher ratios yield more parameters and present a risk of overfitting to the noise in the history of the optimization.

We hypothesize that an MLP-DMD will improve upon the baseline DMD due to the nonlinear approximation. However, it does not use mixing of information between simulation steps, as in our SW-DMD method. This missing attribute of the MLP architecture is because the simulation steps are on the batch dimension.

CNN-DMD. Thus, we also consider a 1D CNN encoder in Figure 2 (b) to form the CNN-DMD method. In CNN-DMD the simulation steps form the temporal dimension and the parameters form

of the CNN. We use causal masking of the CNN kernels on the encoder in order to avoid look-ahead bias. During the inference we look at the history of simulated steps, and look at the prediction of the last step. Then we recurrently feed the last step and resume the predictions. Our architecture consists of two 1D-CNN layers with bottleneck middle channel number of 1 (to avoid overfitting) and ELU activation in between. We hypothesize that the temporal mixing of information via the causal CNN will improve upon MLP-DMD.

MLP-SW-DMD. We further combine the two previous neural approaches into a sliding window approach construction to test the temporal mixing of SW-DMD with the nonlinear approximations of the neural network. Because DMD is a special case of SW-DMD, we also naturally define an MLP-SW-DMD method, i.e. we simply consider our MLP-DMD method for input, obtained by window size larger than 1.

5 EXPERIMENTS

5.1 QUANTUM OPTIMIZATION

For quantum optimization, we perform experiments for the 1D Ising model with the periodic boundary condition and the transverse field h with the goal to find the superposition of spin configurations that minimizes the Hamiltonian $\mathcal{H}$ in Appendix A. We use $h = 0.5$ in this section, and show experimental results for $h = 0.9, 2.0$ in Appendix D. We start from a random initial point in the parameter space to run VQE for n_{sim} iterations. Then, we apply DMD methods to the trajectory of parameters from VQE, predict the future trajectory for n_{DMD} iterations, and evaluate the energies on the predicted trajectory. From the n_{DMD} energy evaluations, we find the minimum energy and the corresponding point in the parameter space, and use this optimal point as the initial point of next n_{sim} iterations of VQE. We repeat the alternating runs of VQE and DMD. n_{sim} and n_{DMD} are hyperparameters in our algorithm. In SW-DMD, the sliding window size n_{SW} is an additional hyperparameter, such that $n_{\text{SW}} < n_{\text{sim}}$. DMD is a special case of SW-DMD with $n_{\text{SW}} = 1$. We benchmark the baseline DMD, and our SW, MLP, MLP-SW and CNN DMDs with $n_{\text{sim}} = 10$. The sliding window size is chosen as $n_{\text{SW}} = 6$ for SW-DMD and MLP-SW-DMD. We also perform a pure VQE run for n_{total} iterations, starting from the same initial point as the VQE+DMD runs.

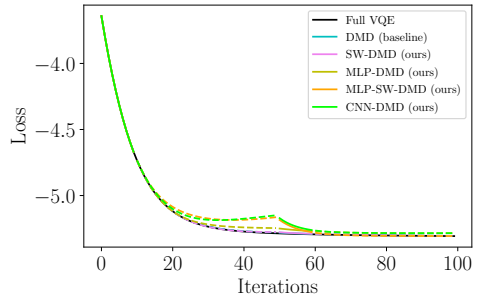
5.1.1 NOISELESS QUANTUM SIMULATIONS

Noiseless quantum Natural gradient simulations. In Figure 3a, in the case of using natural gradient, we compare our DMD methods and the baseline standard DMD method on a 5-qubit Ising model with the circular-entanglement RealAmplitudes ansatz and reps=1 (2 layers, 10 parameters). Solid parts of the lines are from VQE, and dashed lines are from DMD. The learning rate is 0.01. We perform $n_{\text{total}} = 100$ full VQE iterations. Since with the natural gradient, $\boldsymbol{\theta}$ has the quantum wave function as a natural embedding with dynamics close to be linear, Koopman operator learning is hypothesized to yield good prediction. All the DMD algorithms are able to achieve the full VQE optimal loss. DMD and SW-DMD both match the full VQE. Our MLP-DMD, MLPSW-DMD, and CNN-DMD start to deviate from the full VQE when the DMD iterations increase. This could be related to over-fitting of the neural network when the parameters dynamics is simple.

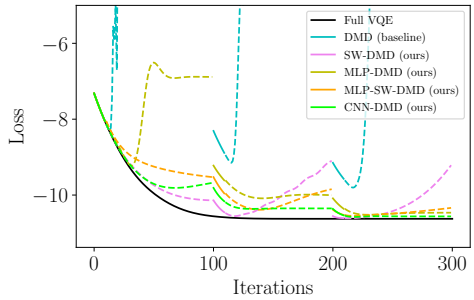
Noiseless Adam simulations. Instead of using the natural gradient, we show another case using the Adam optimizer based on the vanilla gradient obtained from the parameter-shift rule. We perform experiments on the 10-qubit Ising model with the circular-entanglement RealAmplitudes ansatz and reps=1 (2 layers, 20 parameters). In Figure 3b, we choose $n_{\text{total}} = 300$, $n_{\text{sim}} = 10$, $n_{\text{DMD}} = 90$. For SW-DMD and neural SW-DMD, the sliding window size is chosen as $n_{\text{SW}} = 6$. The Adam learning rate is 0.01. DMD has the worst performance, as its lowest loss is still far from the full VQE final loss, whereas all our other methods perform well.

5.1.2 QUANTUM OPTIMIZATION ON A REAL QUANTUM COMPUTER

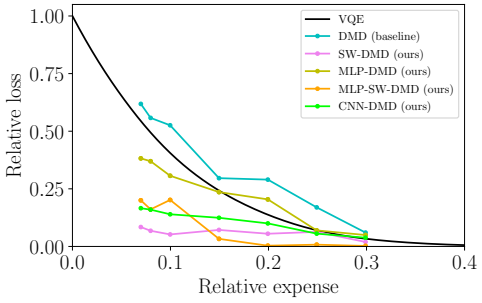
We implement VQE for the 3-qubit Ising model with transverse field $h = 0.5$ on a real IBM quantum computer Lima with the SPSA optimizer (learning rate 0.04 and perturbation 0.1) and 10k quantum shots. We have applied measurement error mitigation to reduce the effect of noise on the real device. We use the RealAmplitudes ansatz with 2 layers (6 parameters) with linear entanglement.



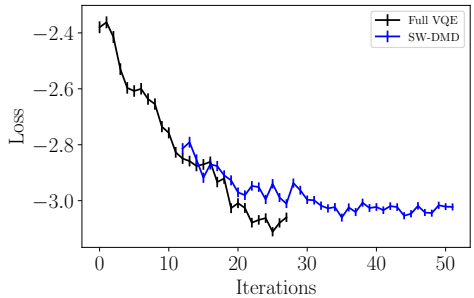
(a) Natural gradient 5-qubit results with $n_{\text{sim}} = 10$, $n_{\text{DMD}} = 40$, and $n_{\text{SW}} = 6$ for SW-DMD and MLP-SW-DMD. For the various DMD methods, solid parts are VQE runs, and the dashed parts are DMD predictions.



(b) Adam 10-qubit results with $n_{\text{sim}} = 10$, $n_{\text{DMD}} = 90$, $n_{\text{SW}} = 6$ for SW-DMD and MLP-SW-DMD. For the various DMD methods, solid parts are VQE runs, and the dashed parts are DMD predictions.



(c) Adam 10-qubit results for relative loss versus relative cost with $n_{\text{sim}} = 7, 8, 10, 15, 20, 25, 30$, $n_{\text{DMD}} = 90$, and $n_{\text{SW}} = 6$ for SW-DMD and neural SW-DMD.



(d) Experimental results from the real quantum computer IBM Lima.

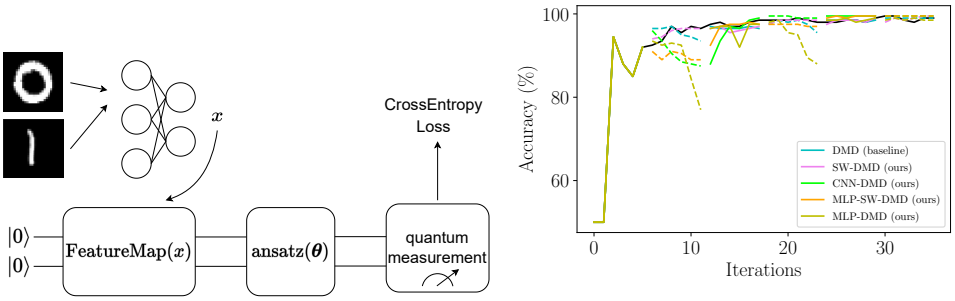
Figure 3: Experimental results.

With the SPSA optimizer, every iteration of update takes 1 center measurement and 2 gradient measurements in only one random direction of the parameter space, so each iteration takes 3 measurements. The smaller number of gradient measurements makes it easier to implement on near-term real quantum computers, compared to using Adam. However,using SPSA may also require a higher number of iterations. The DMD only takes 1 center measurement without gradient measurements.

As is shown in Figure 3d, on IBM Lima, we first perform a full pure VQE for 28 iterations. To mitigate the effect of the random direction in gradient measurements, we use the first 12 VQE iterations for Koopman training, and apply the SW-DMD algorithm with $n_{\text{SW}} = 8$ to make prediction for 40 iterations. In the beginning of prediction, SW-DMD agrees with the full VQE result, and then further decreases the loss although it does not achieve the minimum of the full VQE loss. In this example, the total expense of the first 12 iterations of VQE and 40 iterations of SW-DMD is still smaller than 28 iterations of full VQE. We also note that IBM Lima can have a relatively big uncertainty over different experimental times, and the final difference between the full VQE and SW-DMD could be within the lab experimental uncertainty.

5.2 Quantum Machine Learning

We illustrate an example of Koopman operator learning for QML using a hybrid quantum-classical neural network with its architecture shown in Figure 4a. We consider supervised learning for binary classification over a filtered MNIST dataset with digits 0 and 1. First the MNIST data is fed into a classical neural network for the feature x. Then, the feature x is encoded into the quantum computer via a sequence of quantum gates for feature mapping. Then, the feature-encoded quantum state goes



(a) Quantum machine learning architecture.

(b) Accuracy of binary classification for full QML simulation and DMD methods. For the various DMD methods, solid parts are QML runs, and the dashed parts are DMD predictions.

Figure 4: Quantum machine learning architecture and results.

through an ansatz layer with parameter $\boldsymbol{\theta}$, similar to the case of quantum optimization. We use the results from quantum measurements as the output of the hybrid neural network. Architecture details are in Appendix F. We test our Koopman operator learning algorithms with the above architecture. The full QML training has $n_{\text{total}} = 36$ iterations. We choose $n_{\text{sim}} = 6$, $n_{\text{DMD}} = 6$, and $n_{\text{SW}} = 3$ for SW-DMD and MLP-SW-DMD. Figure 4b shows that DMD methods can achieve reasonably good accuracy while using less quantum resource than the full QML.

5.3 Ablation

With $n_{\text{sim}} = 10$, $n_{\text{DMD}} = 90$, and $n_{\text{SW}} = 6$ for SW-DMD and MLP-SW-DMD, we show an ablation analysis associated by varying n_{sim}. We compute the relative expense spent on the quantum computer measured by the relative number of VQE iterations compared to the full VQE. n_{DMD} is chosen large enough so that this hyperparameter does not affect the minimum loss much. The relative loss is defined as Relative loss $= (L_{\text{min, VQE + DMD}} - L_{\text{min, full VQE}})/(L_{\text{initial, full VQE}} - L_{\text{min, full VQE}})$. A smaller relative loss means better performance. In Figure 3c, we show the relative loss versus the relative expense. The DMD methods should be compared to the performance of pure VQE (black line). All of the DMD methods yield good performance within a tolerance of 10% relative loss, when the relative expense is about 30%, but this is also comparable to the relative loss a pure VQE can achieve. DMD algorithms starts to have advantages over the full VQE only when we further decrease the relative expense by decreasing n_{sim}. The baseline DMD does not have advantage over pure VQE, but all our other DMD methods can perform better than pure VQE when the relative expense is small.

6 Conclusion

Quantum optimization and quantum machine learning have the potential of being powerful for finding solutions of certain problems that are difficult for traditional classical algorithms. However, gradient-based optimization on quantum computers takes $O(n_{\text{params}})$ gradient measurements for each iteration, and thus can be very expensive, especially in the current era where the quantum computing resources are limited. Anchored in a theoretical connection between the Koopman operator theory and the quantum natural gradient, we proposed a data-driven approach to use Koopman learning for accelerating quantum optimization and quantum machine learning. We further used the power of the time-delayed embedding and neural networks to address the nonlinearity in optimization and achieve better prediction which brought down the relative expense of quantum optimization and quantum machine learning both on simulated and real quantum computers. We hope our work serves as a bridge between the machine learning and quantum optimization communities to explore the efficiency of optimization problems through Koopman operator theory.

7 REPRODUCIBILITY OF RESULTS

We have stated all of our hyperparameter choices for the experimental settings in the main text and in the appendix. We perform simulations of VQE using Qiskit (ANIS et al., 2021), a python framework for quantum computation. Our neural network code is based on Qiskit and Pytorch (Paszke et al., 2019). We will release our code on Github upon acceptance.

REFERENCES

Shun-Ichi Amari. Natural gradient works efficiently in learning. *Neural computation*, 10(2):251–276, 1998.

MD SAJID ANIS, Abby-Mitchell, Héctor Abrahamand, et al. Qiskit: An open-source framework for quantum computing, 2021.

Peter J Baddoo, Benjamin Herrmann, Beverley J McKeon, and Steven L Brunton. Kernel learning for robust dynamic mode decomposition: linear and nonlinear disambiguation optimization. *Proceedings of the Royal Society A*, 478(2260):20210830, 2022.

Jacob Biamonte, Peter Wittek, Nicola Pancotti, Patrick Rebentrost, Nathan Wiebe, and Seth Lloyd. Quantum machine learning. *Nature*, 549(7671):195–202, 2017.

Steven L Brunton, Marko Budišić, Eurika Kaiser, and J Nathan Kutz. Modern koopman theory for dynamical systems. *arXiv preprint arXiv:2102.12086*, 2021.

Marco Cerezo, Andrew Arrasmith, Ryan Babbush, Simon C Benjamin, Suguru Endo, Keisuke Fujii, Jarrod R McClean, Kosuke Mitarai, Xiao Yuan, Lukasz Cincio, et al. Variational quantum algorithms. *Nature Reviews Physics*, 3(9):625–644, 2021.

Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*, 2015.

Akshunna S Dogra and William Redman. Optimizing neural networks via koopman operator theory. *Advances in Neural Information Processing Systems*, 33:2087–2097, 2020.

Daoyi Dong, Chunlin Chen, Hanxiong Li, and Tzyh-Jong Tarn. Quantum reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 38(5):1207–1220, 2008.

Daniel Dylewsky, Eurika Kaiser, Steven L Brunton, and J Nathan Kutz. Principal component trajectories for modeling spectrally continuous dynamics as forced linear systems. *Physical Review E*, 105(1):015312, 2022.

S. Ebadi, A. Keesling, M. Cain, T. T. Wang, H. Levine, D. Bluvstein, G. Semeghini, A. Omran, J.-G. Liu, R. Samajdar, X.-Z. Luo, B. Nash, X. Gao, B. Barak, E. Farhi, S. Sachdev, N. Gemelke, L. Zhou, S. Choi, H. Pichler, S.-T. Wang, M. Greiner, V. Vuletić , and M. D. Lukin. Quantum optimization of maximum independent set using rydberg atom arrays. *Science*, 376(6598): 1209–1215, jun 2022. doi: 10.1126/science.abo6587. URL https://doi.org/10.1126%2Fscience.abo6587.

Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A quantum approximate optimization algorithm. *arXiv preprint arXiv:1411.4028*, 2014.

Dimitrios Giannakis, Abbas Ourmazd, Philipp Pfeffer, Jörg Schumacher, and Joanna Slawinska. Embedding classical dynamics in a quantum computer. *Phys. Rev. A*, 105:052404, May 2022. doi: 10.1103/PhysRevA.105.052404. URL https://link.aps.org/doi/10.1103/PhysRevA.105.052404.

Andy Goldschmidt, E Kaiser, J L DuBois, S L Brunton, and J N Kutz. Bilinear dynamic mode decomposition for quantum control. *New Journal of Physics*, 23(3):033035, mar 2021. doi: 10.1088/1367-2630/abe972. URL https://doi.org/10.1088%2F1367-2630%2Fabe972.

Lucas Hackl, Tommaso Guaita, Tao Shi, Jutho Haegeman, Eugene Demler, and Ignacio Cirac. Geometry of variational methods: dynamics of closed quantum systems. *SciPost Physics*, 9 (4), oct 2020. doi: 10.21468/scipostphys.9.4.048. URL https://doi.org/10.21468%2Fscipostphys.9.4.048.

Vojtěch Havlíček, Antonio D. Córcoles, Kristan Temme, Aram W. Harrow, Abhinav Kandala, Jerry M. Chow, and Jay M. Gambetta. Supervised learning with quantum-enhanced feature spaces. *Nature*, 567(7747):209–212, mar 2019. doi: 10.1038/s41586-019-0980-2. URL https://doi.org/10.1038%2Fs41586-019-0980-2.

Hsin-Yuan Huang, Michael Broughton, Jordan Cotler, Sitan Chen, Jerry Li, Masoud Mohseni, Hartmut Neven, Ryan Babbush, Richard Kueng, John Preskill, and Jarrod R. McClean. Quantum advantage in learning from experiments. *Science*, 376(6598):1182–1186, jun 2022a. doi: 10.1126/science.abn7293. URL https://doi.org/10.1126%2Fscience.abn7293.

Hsin-Yuan Huang, Richard Kueng, Giacomo Torlai, Victor V. Albert, and John Preskill. Provably efficient machine learning for quantum many-body problems. *Science*, 377(6613), sep 2022b. doi: 10.1126/science.abk3333. URL https://doi.org/10.1126%2Fscience.abk3333.

Iordanis Kerenidis, Jonas Landman, Alessandro Luongo, and Anupam Prakash. q-means: A quantum algorithm for unsupervised machine learning. *Advances in Neural Information Processing Systems*, 32, 2019.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

N. Klco, E. F. Dumitrescu, A. J. McCaskey, T. D. Morris, R. C. Pooser, M. Sanz, E. Solano, P. Lougovski, and M. J. Savage. Quantum-classical computation of schwinger model dynamics using quantum computers. *Phys. Rev. A*, 98:032331, Sep 2018. doi: 10.1103/PhysRevA.98.032331. URL https://link.aps.org/doi/10.1103/PhysRevA.98.032331.

Stefan Klus, Feliks Nüske, and Sebastian Peitz. Koopman analysis of quantum systems$*$. *Journal of Physics A: Mathematical and Theoretical*, $55(31) : 314002, jul2022. doi :$. URL https://doi.org/10.1088/1751-8121/ac7d22.

B. O. Koopman. Hamiltonian systems and transformation in hilbert space. *Proceedings of the National Academy of Sciences*, 17(5):315–318, 1931. 10.1073/pnas.17.5.315. URL https://www.pnas.org/doi/abs/10.1073/pnas.17.5.315.

Qianxiao Li, Felix Dietrich, Erik M Bollt, and Ioannis G Kevrekidis. Extended dynamic mode decomposition with dictionary learning: A data-driven adaptive spectral decomposition of the koopman operator. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 27(10):103111, 2017.

Yunzhu Li, Hao He, Jiajun Wu, Dina Katabi, and Antonio Torralba. Learning compositional koopman operators for model-based control. *arXiv preprint arXiv:1910.08264*, 2019.

Yen Ting Lin, Robert B. Lowrie, Denis Aslangil, Yiğit Subaşı, and Andrew T. Sornborger. Koopman von neumann mechanics and the koopman representation: A perspective on solving nonlinear dynamical systems with quantum computers, 2022. URL https://arxiv.org/abs/2202.02188.

Yunchao Liu, Srinivasan Arunachalam, and Kristan Temme. A rigorous and robust quantum speed-up in supervised machine learning. *Nature Physics*, 17(9):1013–1017, jul 2021. 10.1038/s41567-021-01287-z. URL https://doi.org/10.1038%2Fs41567-021-01287-z.

Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

Bethany Lusch, J. Nathan Kutz, and Steven L. Brunton. Deep learning for universal linear embeddings of nonlinear dynamics. *Nature Communications*, 9(1), nov 2018. 10.1038/s41467-018-07210-0. URL https://doi.org/10.1038%2Fs41467-018-07210-0.

K. Mitarai, M. Negoro, M. Kitagawa, and K. Fujii. Quantum circuit learning. *Phys. Rev. A*, 98: 032309, Sep 2018. 10.1103/PhysRevA.98.032309. URL https://link.aps.org/doi/10.1103/PhysRevA.98.032309.

Nikolaj Moll, Panagiotis Barkoutsos, Lev S Bishop, Jerry M Chow, Andrew Cross, Daniel J Egger, Stefan Filipp, Andreas Fuhrer, Jay M Gambetta, Marc Ganzhorn, Abhinav Kandala, Antonio Mezzacapo, Peter Müller, Walter Riess, Gian Salis, John Smolin, Ivano Tavernelli, and Kristan Temme. Quantum optimization using variational algorithms on near-term quantum devices. *Quantum Science and Technology*, 3(3):030503, jun 2018. 10.1088/2058-9565/aab822. URL https://doi.org/10.1088/2058-9565/aab822.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library, 2019. URL https://arxiv.org/abs/1912.01703.

Alberto Peruzzo, Jarrod McClean, Peter Shadbolt, Man-Hong Yung, Xiao-Qi Zhou, Peter J. Love, Alán Aspuru-Guzik, and Jeremy L. O'Brien. A variational eigenvalue solver on a photonic quantum processor. *Nature Communications*, 5(1), jul 2014a. 10.1038/ncomms5213. URL https://doi.org/10.1038%2Fncomms5213.

Alberto Peruzzo, Jarrod McClean, Peter Shadbolt, Man-Hong Yung, Xiao-Qi Zhou, Peter J Love, Alán Aspuru-Guzik, and Jeremy L O'brien. A variational eigenvalue solver on a photonic quantum processor. *Nature communications*, 5(1):1–7, 2014b.

Michael JD Powell. A direct search optimization method that models the objective and constraint functions by linear interpolation. In *Advances in optimization and numerical analysis*, pp. 51–67. Springer, 1994.

John Preskill. Quantum Computing in the NISQ era and beyond. *Quantum*, 2:79, August 2018. ISSN 2521-327X. 10.22331/q-2018-08-06-79. URL https://doi.org/10.22331/q-2018-08-06-79.

William T Redman, Maria Fonoberova, Ryan Mohr, Yannis Kevrekidis, and Igor Mezic. An operator theoretic view on pruning deep neural networks. In *International Conference on Learning Representations*, 2021.

Enrico Rinaldi, Xizhi Han, Mohammad Hassan, Yuan Feng, Franco Nori, Michael McGuigan, and Masanori Hanada. Matrix-model simulations using quantum computing, deep learning, and lattice monte carlo. *PRX Quantum*, 3:010324, Feb 2022. 10.1103/PRXQuantum.3.010324. URL https://link.aps.org/doi/10.1103/PRXQuantum.3.010324.

Manuel S. Rudolph, Ntwali Bashige Toussaint, Amara Katabarwa, Sonika Johri, Borja Peropadre, and Alejandro Perdomo-Ortiz. Generation of high-resolution handwritten digits with an ion-trap quantum computer. *Phys. Rev. X*, 12:031010, Jul 2022. 10.1103/PhysRevX.12.031010. URL https://link.aps.org/doi/10.1103/PhysRevX.12.031010.

Peter J Schmid. Dynamic mode decomposition of numerical and experimental data. *Journal of fluid mechanics*, 656:5–28, 2010.

Maria Schuld, Ville Bergholm, Christian Gogolin, Josh Izaac, and Nathan Killoran. Evaluating analytic gradients on quantum hardware. *Phys. Rev. A*, 99:032331, Mar 2019. 10.1103/PhysRevA.99.032331. URL https://link.aps.org/doi/10.1103/PhysRevA.99.032331.

Abhishek Sinha, Mausoom Sarkar, Aahitagni Mukherjee, and Balaji Krishnamurthy. Introspection: Accelerating neural network training by learning weight evolution. *arXiv preprint arXiv:1704.04959*, 2017.

J.C. Spall. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *IEEE Transactions on Automatic Control*, 37(3):332–341, 1992. 10.1109/9.119632.

James Stokes, Josh Izaac, Nathan Killoran, and Giuseppe Carleo. Quantum Natural Gradient. *Quantum*, 4:269, May 2020. ISSN 2521-327X. 10.22331/q-2020-05-25-269. URL https://doi.org/10.22331/q-2020-05-25-269.

Mauricio E. Tano, Gavin D. Portwood, and Jean C. Ragusa. Accelerating training in artificial neural networks with dynamic mode decomposition, 2020. URL https://arxiv.org/abs/2006.14371.

Jules Tilly, Hongxiang Chen, Shuxiang Cao, Dario Picozzi, Kanav Setia, Ying Li, Edward Grant, Leonard Wossnig, Ivan Rungger, George H Booth, et al. The variational quantum eigensolver: a review of methods and best practices. *Physics Reports*, 986:1–128, 2022.

Dave Wecker, Matthew B. Hastings, and Matthias Troyer. Progress towards practical quantum variational algorithms. *Physical Review A*, 92(4), oct 2015. 10.1103/physreva.92.042303. URL https://doi.org/10.1103%2Fphysreva.92.042303.

A Quantum Ising Model with Transverse Field

The quantum Ising model with the periodic boundary condition and transverse field has the Hamiltonian

$$\mathcal{H} = -\sum_{i=1}^{N} Z_i \otimes Z_{i+1} - h \sum_{i=1}^{N} X_i, \tag{13}$$

where

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \tag{14}$$

are Pauli matrices, and the subscripts on them denote which qubit they are acting on. With the periodic boundary condition, the qubits at $i = 1$ and $i = N + 1$ are identical.

B Further details on neural DMD

Activation Along with ELU, we also explored cosine, ReLU and tanh activations. We found ELU to be the best, likely because: tanh suffers from vanishing gradients, ReLU biases to positive numbers (while input phases quantum circuit parameters $\boldsymbol{\theta}$ are unconstrained) and cosine is periodic.

The importance of learning rate scheduler. We train the neural network $\boldsymbol{\Phi}_\alpha$ from scratch every time we obtain optimization history as training data. It is important that we have a stable learning with well converging neural network at every single Koopman operator fitting stage. For that purpose, we found it vital to use a cosine-decay scheduler with a linear warmup, which is typically useful in the computer vision literature (Loshchilov & Hutter, 2016). Namely, for the first 9k steps of the neural network optimization, we linearly scale the learning rate from 0 to 0.001, and then use a cosine-decay from 0.001 to 0 until the final step at 30k.

The importance of residual connections. In our work we use a residual connection in order to make DMD as a special case of the neural DMD parameterization. The residual connection is indeed very useful, as it is driven by the Koopman operator learning formulation. Namely, the residual connection makes it possible for the encoder to learn the identity. If the encoder becomes the identity, then MLP(-SW)-DMD or CNN-DMD become vanilla (SW-)DMD.

C RealAmplitudes Ansatz

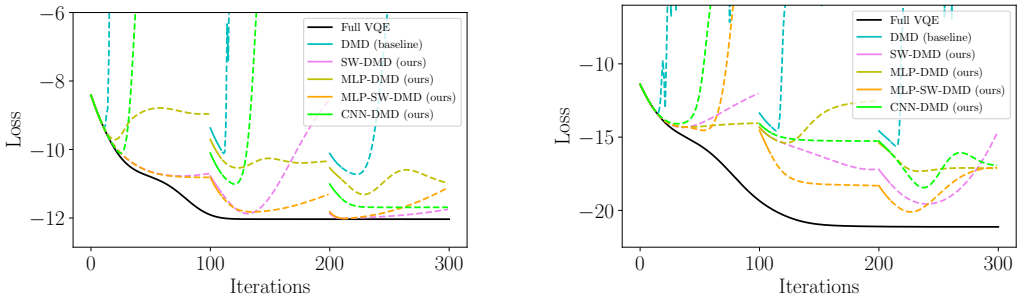
The RealAmplitudes ansatz from Qiskit is an ansatz that always produces a real-valued wave function $\psi_{\boldsymbol{\theta}}$ without the imaginary part. The the minimum loss of Ising-model Hamiltonian is always achievable by a real-valued $\psi_{\boldsymbol{\theta}}$, so this ansatz can reduce the redundancy in the functional form of $\psi_{\boldsymbol{\theta}}$ and is beneficial for our use.

$\psi_{\boldsymbol{\theta}}$ as a function of $\boldsymbol{\theta}$ is a nonlinear function, which consists of alternating layers of the rotational gates for the Pauli matrix

$$Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \tag{15}$$

with rotation angles being the parameters $\boldsymbol{\theta}$, and controlled-X gates with no parameter. Each layer of the rotational gates for Y has N (the number of qubits) parameters, so the number of rotational layers times N is the total number of parameters.

D Results at More Values of h



(a) $h = 0.9$.

(b) $h = 2.0$.

Figure 5: 10-qubit noiseless Adam simulation results with $n_{\text{sim}} = 10$, $n_{\text{DMD}} = 90$, $n_{\text{SW}} = 6$, $n_{\text{total}} = 300$ for SW-DMD and MLP-SW-DMD at (a) $h = 0.9$ and (b) $h = 2.0$. For the various DMD methods, solid parts are VQE runs, and the dashed parts are DMD predictions.

We show the 10-qubit Adam results at $h = 0.9, 2.0$ in Figure 5 with the same setup as we do for $h = 0.5$. The various DMD methods lower the loss, but some of them still need more iterations to get closer to the minimum loss of full VQE.

E Comparison between Real Lima and FakeLima

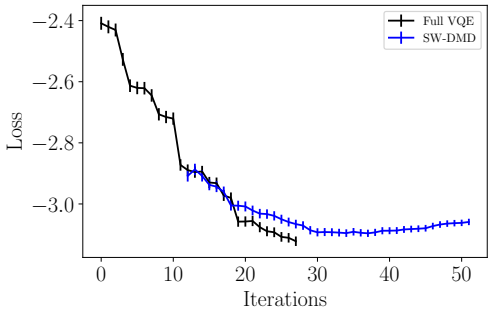


Figure 6: FakeLima results.

We perform simulation using FakeLima, a noise model designed to mimic the real Lima, provided in Qiskit on the same 3-qubit Ising model at $h = 0.5$ as we do on the real Lima. We use the same setup as what we use on the real Lima with $n_{\text{total}} = 28$, $n_{\text{sim}} = 12$, $n_{\text{SW}} = 8$ with SW-DMD. The FakeLima results are shown in Figure 6, and are qualitatively consistent with Figure 3d with less fluctuation in the SW-DMD loss evaluation.

F QML ARCHITECTURE DETAILS

The classical neural network in the hybrid architecture consists of [Conv2D(in-channel=1, out-channel=2, kernel-size=5), ReLU(), MaxPool2D(size=2), Conv2D(in-channel=2, out-channel=2, kernel-size=5), ReLU(), Linear(in-dim=2, out-dim=2), ReLU(), Linear(in-dim=2, out-dim=2)].

The feature mapping is realized through the ZZFeatureMap on two qubits with one repetition from Qiskit. The ansatz after the feature mapping is the RealAmplitude ansatz with one repetition. The quantum measurements are performed on the Z-basis of two qubits, which return the expectation value of $\langle Z_1 Z_2 \rangle$ as an output y. Then, y and $1-y$ are used as logits for the CrossEntropy loss. Adam is used as the optimizer with learning rate 5×10^{-4}. 50 training examples and 100 test examples are used for each digit.