




Protocols for trainable and differentiable quantum generative modeling

Oleksandr Kyriienko ^{1,2}, Annie E. Paine ^{1,2} and Vincent E. Elfving ²

¹*Department of Physics and Astronomy, University of Exeter, Stocker Road, Exeter EX4 4QL, United Kingdom*

²*Pasqal SAS, 2 av. Augustin Fresnel, 91120 Palaiseau, France*



(Received 18 August 2023; accepted 6 August 2024; published 12 September 2024)

We propose an approach for learning probability distributions as differentiable quantum circuits (DQC) that enable efficient quantum generative modeling (QGM) and synthetic data generation. Contrary to existing QGM approaches, we perform training of a DQC-based model, where data is encoded in a latent space with the proposed phase feature map of exponential capacity. This is followed by a trainable quantum circuit, forming the model. We then map the trained model to the bit basis using a fixed unitary transformation, in this case corresponding to a quantum Fourier transform circuit. It allows fast sampling from parametrized distributions using a single-shot readout. Importantly, latent space training provides models that are automatically differentiable, and we show how samples from solutions of stochastic differential equations (SDEs) can be accessed by solving stationary and time-dependent Fokker-Planck equations with a quantum protocol. Our approach opens a route to multidimensional generative modeling with qubit registers explicitly correlated via a (fixed) entangling layer. In this case quantum computers can offer advantage as efficient samplers, which perform complex inverse transform sampling enabled by the fundamental laws of quantum mechanics. On a technical side the advances are multiple, as we introduce the phase feature map, analyze its properties, and develop frequency-taming techniques that include qubitwise training and feature map sparsification.

DOI: [10.1103/PhysRevResearch.6.033291](https://doi.org/10.1103/PhysRevResearch.6.033291)

I. INTRODUCTION

Quantum computing (QC) promises to offer a computational advantage by meticulous usage of an exponentially large Hilbert space for qubit registers [1]. However, the use of QC is limited to specific tasks, as efficient solutions are only expected for some problem types [2]. One example corresponds to sampling from quantum states created by random entangling circuits [3,4]. This task lies at the heart of quantum supremacy experiments [5–8]. While being computationally advantageous for producing samples (just need to send a “measure” instruction), the considered distributions are not suitable for industrially relevant advantage [9], though may be helpful in studying related concept such as quantum chaos [10]. Finding a subset of problems with distributions which are both classically intractable and industrially useful is an open challenge. Quantum generative modeling (QGM) aims to exploit trainable circuits that can prepare distributions as quantum states, for instance, trying to match patterns from available data. Being a subject of the emerging field of quantum machine learning (QML) [11,12], QGM utilizes the Born rule inherent to quantum mechanics [13]. The goal is to represent a parametrized probability distribution $p_{\theta}(x)$. It represents a probability to measure a bit string x from a variational state $|\psi_{\theta}\rangle$ parametrized by a vector of gate parameters θ [14,15]. For the simple case of pure

states this reads $p_{\theta}^{\text{QCBM}}(x) = |\langle x|\psi_{\theta}\rangle|^2$. This approach is the basis of quantum circuit Born machines (QCBMs) [16] that learn models directly from samples of a target distribution $p_{\text{target}}(x)$ using various implicit loss functions [17,18]. A similar approach is used for generating circuits in quantum generative adversarial networks (QGANs) [19–23], where the training schedule corresponds to the minimax game. To date, QCBMs have been used for loading static distributions corresponding to bars-and-stripes dataset [15,17], learning datasets of correlated currency pairs [24], and digitized Gaussian and bimodal distributions [17]. QGANs were used for (reduced) MNIST datasets [25], financial modeling [20], learning pure states [26], and sampling particle traces [27]. While making a step toward sampling-based advantage, current QGM performance remains limited even for idealized statevector simulators [17], and in the case of QCBM may require matching all state amplitudes for representing particular distributions. Developing models that can balance expressivity and trainability is an open problem.

A promising application for generative modeling corresponds to solving stochastic differential equations (SDEs), where system evolves in the presence of noise, and expected behavior relies on averaging over multiple samples and trajectories [28]. Note that the samples are generated from an underlying probability distribution, which obeys a Fokker-Planck equation. Depending on the problem context, this may also require sampling from evolving distributions, and assuming differential constraints added into the generative model. However, many popular generative modeling algorithms are not suitable for this task. For example, QCBM architecture is not automatically differentiable with respect to variable x , and QGAN differentiation leads to an ill-defined loss landscape

Published by the American Physical Society under the terms of the [Creative Commons Attribution 4.0 International license](https://creativecommons.org/licenses/by/4.0/). Further distribution of this work must maintain attribution to the author(s) and the published article's title, journal citation, and DOI.

[29]. Thus, both have limited application for SDE solving. The latter would be hugely beneficial as differential constraints remove strong dependence on data, regularize models, and offer additional structure to learning (see quantum approach to adding differential constraints in Refs. [30,31] and physics-informed neural network architectures in classical machine learning [32–34]). SDE-based sampling is also motivated by works in the financial sector where Monte Carlo techniques are used. To date, various quantum protocols for associated PDEs has been considered, in many cases taking the perspective of real and imaginary time evolution [35–38] or using amplitude amplification for tasks like option pricing [39–43]. More broadly, the area of differential equations with quantum computers has been developing rapidly, starting from fault-tolerant QC oriented [44–47] to near-term and quantum-inspired protocols [30,31,48–52]. Furthermore, differentiable distributions allow for the use of gradient ascent which enables extremal learning [53], with relevant applications in design/optimization tasks.

We first note that the ability of differentiating generative models can be restored when using feature map encoding of continuous distributions [54], at the expense of multishot measurement to get a sample from QNNs. Second, the differential constraints at the sampling stage can be implemented using quantum quantile mechanics (QQM) [29], where a quantum circuit is trained to generate samples from SDEs and can be evolved in time, albeit with expectation-based sampling. Here, merging differentiability with fast sampling will offer both potential expressivity advantage and sampling advantage of QC.

In this work we develop a workflow for training of quantum generators that can be differentiated with respect to a continuous stochastic variable. For this, we separate the training and sampling stages of QGM. During the training stage we build an explicit model for probability distributions in the latent space (taken as a phase space) enabled by the *phase feature map*, followed by a variational circuit, and DQC-type readout. The sampling stage is then performed in the bit basis space enabled by the fixed unitary transformation (quantum Fourier transform), and followed by projective measurements for a sample-by-sample readout. The proposed workflow leads to *differentiable* quantum generative models (DQGM [55]), and is used for sampling from SDEs. Another consequence of training in the phase space is inherent model regularization, enforced by the proposed *qubitwise* learning, *feature map sparsification*, and *frequency-taming* techniques for circuit initialization based on Fourier series. Showing probability distribution (or generic function) loading into state amplitudes, we proceed to solve Fokker-Planck equations, giving access to time-series of the Ornstein-Uhlenbeck process. Finally, considering correlated registers where quantum correlations are included by entangling circuits [56,57], we discuss how classically hard multidimensional distributions can be automatically “inverted” by QCs, making a step toward a sampling advantage.

II. THE APPROACH

Generative modeling concerns the process of drawing samples of a stochastic variable $X_t \sim p_{\theta,t}(x)$ from a trainable distribution with variational angles θ , which is also

parametrized by t . Typically, we associate t to time as a deterministic variable, which may enter explicitly (as an additional parameter) or implicitly encoded in $\theta(t)$. We will use the notation θ, t throughout for both cases, and specify encoding where ambiguity may arise. In the generic quantum case the model can be constructed using Born’s rule, $p_{\theta,t}(x) = \text{tr}\{|x\rangle\langle x|\hat{\rho}_{\theta,t}\}$, where samples x corresponding to length- N binary strings are readout from the density operator $\hat{\rho}_{\theta,t} = \mathcal{E}_{\theta,t}(\hat{\rho}_0)$ created by a parametrized completely positive trace-preserving (CPTP) map $\mathcal{E}_{\theta,t}$ from some initial density operator $\hat{\rho}_0$. The latter typically corresponds to the computational zero state $\hat{\rho}_0 = |\phi\rangle\langle\phi|$, where $|\phi\rangle \equiv |0\rangle^{\otimes M}$ for $M \geq N$. In many cases unitary quantum channels are considered, $\mathcal{E}_{\theta,t}(\hat{\rho}_0) = \hat{U}_{\theta,t}\hat{\rho}_0\hat{U}_{\theta,t}^\dagger$ with $M = N$ and $\hat{U}_{\theta,t}$ is a generic parametrized unitary on N -qubit register. Note that when $\hat{U}_{\theta,t} \in \mathcal{SU}(2^N)$ in principle any state of the register can be prepared, and we call such a model maximally expressive. We recall that typically QCBM-style generative modeling relies on sample-based training of $p_{\theta,t}^{\text{QCBM}}(x) = \text{tr}\{|x\rangle\langle x|\hat{U}_{\theta,t}\hat{\rho}_0\hat{U}_{\theta,t}^\dagger\}$ at digital (i.e., integer, binary) values of x only, and angles θ are sought separately at different points of time t . The generic goal is minimizing a loss function $\mathcal{L}_{\theta,t}^{\text{QCBM}} = \sum_{x=0}^{2^N-1} \mathcal{D}[p_{\text{target}}(x,t), p_{\theta,t}^{\text{QCBM}}(x)]$, for some distance measure $\mathcal{D}[\cdot, \cdot]$. The optimization procedure gives the optimal angles $\theta_{\text{opt}} = \text{argmin}_{\theta}[\mathcal{L}_{\theta,t}^{\text{QCBM}}]$ at fixed t . In practice, this is achieved using data samples $x \in \mathcal{X}_{\text{data}}$ (typically, from observations) and a proxy loss, corresponding to maximum mean discrepancy (MMD) [17], Stein discrepancy (SD) [18], Kullback-Leibler divergence, as well as other types of f-divergences [58]. Once $p_{\theta,t}^{\text{QCBM}}(x)$ is successfully trained, one can proceed directly to sampling from the same circuit.

We propose to act differently. We start by describing the protocol for generating computational states $\{|x\rangle\}$ (each associated to binary strings $x \in \mathcal{B} = \{00..0, 10..0, \dots, 11..1\}$). This can be achieved in two steps. First, a parametrized feature map creates a latent (phase) space representation of the variable x , $\hat{\rho}_{\tilde{x}} = \hat{U}_{\varphi}(x)\hat{\rho}_0\hat{U}_{\varphi}^\dagger(x)$. For convenience, we call the corresponding circuit the *phase feature map*. For $\hat{\rho}_0 = |\phi\rangle\langle\phi|$ it reads

$$\hat{U}_{\varphi}(x) = \prod_{j=1}^N \left[\hat{R}_j^z \left(\frac{2\pi x}{2^j} \right) \hat{H}_j \right], \quad (1)$$

where $\hat{R}_j^z(\phi) = \cos(\phi/2)\hat{1}_j - i\sin(\phi/2)\hat{Z}_j$ is a single-qubit rotation and \hat{H}_j is a Hadamard gate, acting at site j . The circuit in Eq. (1) maps an initial state into a superposition *product* state $\hat{\rho}_{\tilde{x}} = |\tilde{x}\rangle\langle\tilde{x}|$ based on the latent state $|\tilde{x}\rangle := \hat{U}_{\varphi}(x)|\phi\rangle$, which explicitly reads

$$|\tilde{x}\rangle = \frac{e^{-i\Phi/2}}{2^{N/2}} \bigotimes_{j=1}^N \left(|0\rangle_j + \exp\left(-i\frac{2\pi x}{2^j}\right) |1\rangle_j \right), \quad (2)$$

where $\Phi = 2\pi(1 - 2^{-N})$ is an overall phase. Importantly, the phase space representation contains all computational basis states, which we can label by integers $\{x_\ell\} = \{0, 1, \dots, 2^N - 1\}$, and associated states are not entangled. Next, we apply the quantum circuit \hat{U}_{T} such that it transforms latent states $\{|\tilde{x}_\ell\rangle\}$ into binary states $\{|x\rangle\}$ as a bijection. The subscript φ highlights that the transformation circuit is designed for

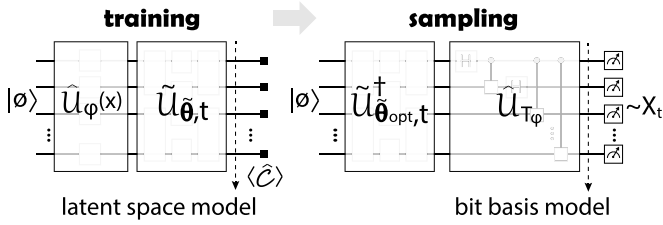


FIG. 1. DQGM training and sampling. At the training stage we use the latent space model representation, where the phase feature map directly follows by the variational circuit (and basis transformation circuits effectively cancel each other). At the sampling stage, we revert the trained variational circuit and map the model from the latent to the bit space, while the feature map and inverse basis transformation are treated as a part of the projective measurement, and are subsumed in a sampling process.

the specific feature map. The corresponding density operator $\hat{\rho}_x = \hat{U}_{T_\varphi} \hat{\rho}_x \hat{U}_{T_\varphi}^\dagger$ thus encodes the variable x in the bit basis. We note that in the considered case such transformation represents an inverse quantum Fourier transform circuit, $\hat{U}_{T_\varphi} = \hat{U}_{\text{QFT}}^\dagger$ [59], which consists of $O(N^2)$ gates (Hadamards and controlled-phase). Other options are available for different basis choices, and we discuss them in the conclusions section. Having generated the state $\hat{\rho}_x$ we proceed by applying a variational ansatz. We choose it in the form $\hat{W}_{\tilde{\theta},t} = \hat{U}_{\tilde{\theta},t} \hat{U}_{T_\varphi}^\dagger$, where with the tilde in $\tilde{\theta}$ and $\tilde{U}_{\tilde{\theta},t}$ we highlight that the circuit structure and parametrization angles are different from QCBM. Our strategy is building a differentiable quantum generative model (DQGM [55]), fully in the latent space, $\tilde{p}_{\tilde{\theta},t}(x) = \text{tr}\{\hat{C}_\phi \tilde{U}_{\tilde{\theta},t} \hat{\rho}_x \tilde{U}_{\tilde{\theta},t}^\dagger\}$, with the cost (measurement) operator being $\hat{C}_\phi \equiv \rho_0$. The model is trained to match the target distribution for $\theta_{\text{opt},t} = \text{argmin}_{\tilde{\theta}} \sum_{x \in \mathcal{X}} \mathcal{D}[p_{\text{target}}(x, t), \tilde{p}_{\tilde{\theta},t}(x)]$ for a grid \mathcal{X} of real-valued $x \in [0, 2^N - 1)$ (or in other normalized interval), at given t . Note that due to training in the latent space the cost can be also a local operator [60], or single-ancilla SWAP/Hadamard test for measuring the overlap. We then sample the trained model using projective measurements as $X_t \sim p_{\tilde{\theta}_{\text{opt},t}} = \text{tr}\{|x\rangle\langle x| \hat{U}_{T_\varphi} \hat{U}_{\tilde{\theta}_{\text{opt},t}}^\dagger \hat{\rho}_0 \tilde{U}_{\tilde{\theta}_{\text{opt},t}} \hat{U}_{T_\varphi}^\dagger\}$ (see Fig. 1). To show that we can sample the model successfully in the bit basis, let us formulate the connection between DQGM and QCBM in Theorem 1 below.

Theorem 1. Probability distributions of binary samples $\{X_t\}$ from maximally expressive QCBM at global optimum θ_{opt} and maximally expressive DQGM at global optimum θ_{opt} are equivalent.

Proof. Generative modeling from QCBM can be expressed as sampling from a generalized probability distribution

$$p_{\theta,t}^{\text{gQCBM}}(x) = \text{tr}\{|x\rangle\langle x| \hat{U}_{\theta,t} \hat{\rho}_0 \hat{U}_{\theta,t}^\dagger\} \quad (3)$$

$$= \text{tr}\{\hat{C}_\phi \hat{U}_\varphi^\dagger(x) \hat{U}_{T_\varphi} \hat{U}_{\theta,t} \hat{\rho}_0 \hat{U}_{\theta,t}^\dagger \hat{U}_{T_\varphi} \hat{U}_\varphi(x)\}, \quad (4)$$

where $\hat{U}_\varphi^\dagger(x)$ corresponds to the phase feature map. At digital values of the variable Eq. (4) corresponds to $p_{\theta,t}^{\text{QCBM}}(x)$, but extends QCBM to $x \in \mathbb{R}$. Note that in the intervals between digital points $\ell < x < \ell + 1$ ($\ell = 0, 1, \dots, 2^N - 2$) the samples come from the superposition of neighboring states,

$\propto \alpha|x_\ell\rangle + \beta|x_{\ell+1}\rangle$ (with x -dependent complex coefficients α, β), preserving sampling locality. The latent DQGM model can be rewritten as

$$\tilde{p}_{\tilde{\theta},t}(x) = \text{tr}\{\hat{\rho}_x \tilde{U}_{\tilde{\theta},t}^\dagger \hat{\rho}_0 \tilde{U}_{\tilde{\theta},t}\} = \text{tr}\{|x\rangle\langle x| \hat{W}_{\tilde{\theta},t}^\dagger \hat{\rho}_0 \hat{W}_{\tilde{\theta},t}\}, \quad (5)$$

directly following from cyclic properties of the trace and previously introduced definitions. Comparing models in Eqs. (3) and (5), and given that quantum states $\hat{U}_{\theta,t} \hat{\rho}_0 \hat{U}_{\theta,t}^\dagger$ and $\hat{W}_{\tilde{\theta},t}^\dagger \hat{\rho}_0 \hat{W}_{\tilde{\theta},t}$ are trained to match the same target distribution, for maximally expressive circuits $\hat{U}_{\theta,t}, \tilde{U}_{\tilde{\theta},t} \in SU(2^N)$ the probability distributions match at the global optimum, $p_{\theta_{\text{opt},t}}^{\text{gQCBM}}(x) = \tilde{p}_{\tilde{\theta}_{\text{opt},t}}(x)$. This follows from the fact that both circuits are *in principle* capable of expressing any state (quasidistribution) [59,61], where $\hat{W}_{\tilde{\theta},t}$ can absorb a fixed transformation by readjusting the angles, and both aim to prepare the same optimal state. ■

While we show that the two approaches are equivalent during the sampling stage, the two models are vastly different during the training stage. For the QCBM and its generalization in Eq. (4) the sampling and training settings are the same. They require a variational state to match bit string probabilities already in training. This basis may work better for peaked or discrete distributions (like bars-and-stripes), but challenging for smooth functions which are continuous and have no sharp transitions (equivalently, have large filling ratio [62]). For the DQGM we only require training of the latent model, where a superposition product state is obtained from x -parametrized single qubit rotations (spans all $O(2^N)$ amplitudes) and needs a certain overlap with a variational state (with a support of the same size). Intuitively, this task is easier to achieve, and we substantiate the claim later when conducting numerical experiments. As DQGM and QCBM originate from the same phase feature map, they have the same *model capacity*—spectrum characterized by exponentially large number of frequencies (considered in the next subsection). At the same time, DQGM has better *model expressivity* in terms of access to Fourier coefficients for relevant low-frequency components, thanks to the (nonvariational) unitary transformation \hat{U}_{T_φ} that can remove a part of the training complexity. Due to all of this our approach is more suitable for smooth models.

1. Model differentiation and constrained training from stochastic differential equations

One of the important consequences of the proposed approach is the possibility for differentiating a constructed quantum model. This can be done by using quantum automatic differentiation (AD) applied to the phase feature map [30]. Note that as we use the latent model in training, we can apply differential constraints already at this stage. Only once trained we proceed to sampling. Let us discuss examples where such physics (or finance/biology/chemistry) constraints are important. Consider a stochastic differential equation written as [28]

$$dX_t = f(X_t, t)dt + g(X_t, t)dW_t, \quad (6)$$

where dW_t is a standard Wiener process, X_t is time-dependent stochastic variable, and $f(\cdot), g(\cdot)$ are some scalar functions

typically referred as drift and diffusion. For any SDE in the form of Eq. (6) we can write an equation of motion for the probability distribution. This can correspond to a Fokker-Planck equation (FPE)

$$\frac{\partial}{\partial t} p(x, t) = - \frac{\partial}{\partial x} [f(x, t)p(x, t)] + \frac{1}{2} \frac{\partial^2}{\partial x^2} [g^2(x, t)p(x, t)], \quad (7)$$

written for the time-dependent probability distribution function $p(x, t)$ of the stochastic variable X_t . This is the PDE which describes the evolution of $p(x, t)$ from a known initial distribution at a time t_0 under the random forces (noise) as stated in Eq. (6). The FPE is a special case of the Kolmogorov Forward equation. Alternatively, if instead we want to determine previous distributions for a given end distribution, then a Kolmogorov backward equation (KBE) [63] is used. For the case of Eq. (6) this can be written as

$$- \frac{\partial}{\partial t} p(x, t) = f(x, t) \frac{\partial}{\partial x} p(x, t) + \frac{g^2(x, t)}{2} \frac{\partial^2}{\partial x^2} p(x, t). \quad (8)$$

This is solved for $p(x, t)$, $t \leq T$ where $p(x, T)$ is a known final distribution.

More generally, the evolution can be described by the Feynman-Kac formula [38]. Importantly, once we learn the $p(x, t)$ in the domain of interest $t \in \mathcal{T}$, in-principle we can obtain stochastic trajectories (samples from time-incremented distributions), offering full generative modeling of time-series. Normally, given only access to a function $p(x, t)$, generating samples requires a costly inversion procedure (or equivalent), and is challenging for multidimensional problems. For the quantum generative models it requires learning t -parametrized DQGM at different times, giving direct access to fast sampling. Below we sketch the workflow, and provide more details when considering examples in the Results section.

The stochastic problem (6) can be approached from a data-driven perspective, where we first learn a representation of the steady state from available samples. This is highly relevant also from the point of view of model discovery [64], as drift and diffusion coefficients may not be immediately known. Setting the loss function for DQGM as $\mathcal{L}_{\theta, t_0}^{\text{data}} = \sum_{x \in \mathcal{X}} \mathcal{D}[p_{\text{target}}(x, t_0), \tilde{p}_{\theta, t_0}(x)]$, we can learn a distribution at a point of time t_0 .

Now, let us comment on two possible ways of encoding the time variable. First, time t can be embedded explicitly. One option is to use a t -dependent feature map for parametrizing the model. For instance, we employed it successfully in DQC-based quantum function propagation [29]. In this case, it is convenient to use an identity-valued feature map at t_0 , and learn to adjust angles as t deviates from t_0 . Second, explicit encoding of time can take a polynomial of t (or even a feed-forward neural network), with θ 's being trainable coefficients. In this case, $t = t_0$ training can be performed for zeroth degree term, and adjusting remaining coefficients at other times. Finally, we can also assume an implicit dependence of variational coefficients $\theta(t)$ on time. In this case, we learn to represent data at t_0 with parameters $\theta(t_0)$, and then demand that each point of time the distribution satisfies differential constraints for a PDE in question. This leads to model-dependent updates of variational parameters

$\theta(t + \Delta t) \stackrel{\gamma}{\leftarrow} \theta(t)$ (with an update rule γ), thus evolving the model in discrete time [65]. Below, we show how to introduce model-dependent differential constraints, and training or evolving DQGM in both explicit and implicit manner. We note both are physics-informed, and represent a step forward from static sample generation.

Given the SDE in Eq. (6), the evolution of associated $p(x, t)$ requires solving a PDE either forward or backward in time. The former case corresponds to solving the Fokker-Planck equation (corresponding to the Kolmogorov forward equation) written in Eq. (7). We evolve the system toward the stationary state at $t_s > t$ from some initial distribution. The stationary distribution of FPE then satisfies the second-order differential equation

$$\text{FPE}(p, x, t_s; f, g) := - \frac{d}{dx} [f(x, t_s)p(x, t_s)] + \frac{1}{2} \frac{d^2}{dx^2} [g^2(x, t_s)p(x, t_s)] = 0, \quad (9)$$

and we call the corresponding differential constraint on the distribution the FPE differential operator. Specifically, we can substitute $p(x, t_s)$ with $\tilde{p}_{\theta, t_s}(x)$ and train a quantum generative model to respect the FPE constraint assigning the differential loss $\mathcal{L}_{\theta, t_s}^{\text{diff}} = \sum_{x \in \mathcal{X}} \mathcal{D}[0, \text{FPE}(\tilde{p}_{\theta, t_s}, x; f, g)]$, such that it remains true for all x . We note that this inherently regularizes the model, and in particular leads to improved derivative matching, highly relevant for studying tails of distributions and dynamics.

Next, we note that we can train a quantum model to represent the PDF at some point of time t_0 , using data as a snapshot during evolution. Then, the full PDE and associated differential constraints are used to propagate it in the $t_0 < t < t_s$ interval reaching the steady state at t_s . Specifically, we can write the differential loss based on the difference of the RHS and the LHS of the FPE, which we call the dynamical FPE differential operator $\text{DFPE}(p, x, t; f, g)$. The loss dictates that our model minimizes $\mathcal{L}_{\theta}^{\text{evol}} = \sum_{x, t \in \mathcal{T} \times \mathcal{X}} \mathcal{D}[0, \text{DFPE}(\tilde{p}_{\theta, t}, x; f, g)]$, and we assume explicit time embedding. Then the workflow for evolving differentiable quantum generative models has a style similar to PINN/DQC workflow [30]. Once done, the model can be sampled within the trained region, and generalized in between the points.

Alternatively, we can use an evolutionary approach for updating circuit parameters [65]. In this case, the time-derivative of our model $\partial \tilde{p}_{\theta, t}(x) / \partial t$ can be reexpressed using a chain rule as $(\partial \tilde{p}_{\theta, t}(x) / \partial \theta) (\partial \theta / \partial t)$. The differential constraints in space and time then require that a vector of updates satisfies $\gamma = (\mathbf{J}^T \cdot \mathbf{J})^{-1} \cdot \mathbf{J}^T \cdot \mathbf{F}$, where \mathbf{F} is a vector corresponding to differential operator $\text{FPE}(\tilde{p}_{\theta, t}, x; f, g)$ evaluated at the grid $x \in \mathcal{X}$. The matrix \mathbf{J} is the Jacobian for our model evaluated at $x \in \mathcal{X}$, each having $|\theta|$ entries. The update can be performed using a simple Euler's forward update $\theta(t + \Delta t) = \theta(t) + \Delta t \gamma$, where Δt is a time step, and we stress that γ is recalculated as we "march" over the grid of times. Going beyond linear updates, more sophisticated schemes (e.g., Runge-Kutta) can be employed.

Finally, we can evolve the probability distribution using the Kolmogorov backward equation (KBE), where the goal is to study the dynamics at times prior to the steady state. Let us

define $\tau < t_s$ as a backward time. A generic KBE associated to the SDE in Eq. (6) is written in Eq. (8). It is convenient to set a starting point $\tau = t_s$ and find $p(x, \tau < t_s)$ backward in time, discovering (and sampling) the model at earlier times. All steps discussed before apply here as well.

Once we define the setting for solving problems based on SDE/PDE, we need to specify how to differentiate the proposed model (something that is not possible with QCBM/QGAN architectures). In the next subsection, where we analyze the phase feature map, we will also show how to read out x derivatives of DQGM. While this can be done through the parameter shift rule [66,67] and generalizations [68], can be readout exactly and more efficiently by avoiding the regular parameter shift rule.

A. Phase feature map analysis

We note that by construction the latent space probability distribution $\tilde{p}_\theta(x)$ corresponds to a parametrized quantum circuit with feature map encoding [69–72], and can be analyzed by studying associated Fourier series (for brevity, we omit t dependence in this subsection). We proceed to analyze the model capacity of the phase feature map $\hat{U}_\varphi(x)$. While Chebyshev series are available with additional variable transformations [30], for the discussed phase map with homogeneous we remain in Fourier space (potential upgrades to other basis sets are discussed in the end of the paper). Specifically, we define capacity as the number of modes (frequencies) that are *in principle* available in the model. This is determined by the spectral properties of the generator of the feature map, $\hat{G} : \hat{U}_\varphi(x) = \exp(-ix\hat{G}/2)$. We note that parametrized quantum circuits can generally represent a function (model) as

$$f_\theta(x) = \sum_{\omega \in \Omega} c_{\omega, \theta} e^{i\omega x}, \quad (10)$$

where the spectrum of frequencies Ω represent all possible *differences* of eigenvalues of \hat{G} , and $c_{\omega, \theta}$ are θ -dependent coefficients associated to each frequency [68,71]. The important properties of the spectrum are that it includes zero frequency, pairs of equal-magnitude positive and negative frequencies, and coefficients obey $c_\omega = c_{-\omega}^*$ leading to real-valued models (as expected from an expectation value). While the analysis can proceed by studying the generator of the phase map, here we derive model capacity explicitly from the latent state written in Eq. (2). Let us define the phase for each qubit rotation as $\varphi_j := 2\pi/(2^j)$. The N -qubit superposition state $|\tilde{x}\rangle$ has an equal overlap with all computational basis states, $|\langle x|\tilde{x}\rangle|^2 = 1/2^N \forall x \in \mathcal{B}$, but each individual contribution comes with a different phase (sum of individual φ_j 's). Expanding the tensor product in Eq. (2) we see that the computational zero state $|\phi\rangle$ has a phase of zero, by convention. Next, there are N states with single excitations, $|j\rangle := e^{i\varphi_j x} \hat{X}_j |\phi\rangle$, each with their phase exponentially decreasing from the highest ($\varphi_1 = 2\pi/2$) to lowest ($\varphi_N = 2\pi/2^N$) as qubit number increases. Next, we have $N(N-1)/2$ states with double excitations, $|jj'\rangle := e^{i(\varphi_j + \varphi_{j'})x} \hat{X}_j \hat{X}_{j'} |\phi\rangle$, with corresponding phases of a sum of contributions. In general, there are $N!/m!(N-m)!$ states with m

excitations (and sums of m phases), culminating with a fully excited state $|\mathbb{1}\rangle := e^{i\Phi} \hat{X}^{\otimes N} |\phi\rangle$, with $\Phi = \sum_j \varphi_j = 2\pi(2^N - 1)/2^N$. We collect sum of phases associated to bit basis states $\{|x_\ell\rangle\}$, calling them frequencies $\{v_\ell\} = \{2\pi \ell/2^N\}_{\ell=0}^{2^N-1}$ at this point. We note that the latent state can be rewritten in a simple form $|\tilde{x}\rangle = (e^{-i\Phi/2}/2^{N/2}) \sum_{\ell=0}^{2^N-1} e^{iv_\ell x} |x_\ell\rangle$. Next, we proceed to construct the model itself as in Eq. (5), which comes from the overlap (squared) of the latent feature state with an ansatz-prepared state, $\hat{U}_\theta |\phi\rangle = \sum_{\ell=0}^{2^N-1} a_{\ell, \theta} |x_\ell\rangle$ (hereafter we simplify the notation by removing tildes where appropriate). The latent space probability distribution then reads

$$\begin{aligned} \tilde{p}_\theta(x) &= \frac{1}{2^N} \sum_{\ell, \ell'=0}^{2^N-1} a_{\ell, \theta}^* a_{\ell', \theta} e^{i(v_\ell - v_{\ell'})x} \\ &= \frac{1}{2^N} + \frac{1}{2^{N-1}} \sum_{\ell > \ell'} \{ \text{Re}\{a_{\ell, \theta}^* a_{\ell', \theta}\} \cos[(v_\ell - v_{\ell'})x] \\ &\quad - \text{Im}\{a_{\ell, \theta}^* a_{\ell', \theta}\} \sin[(v_\ell - v_{\ell'})x] \}, \end{aligned} \quad (11)$$

where in the second and third line of Eq. (11) we split the double sum to show real and imaginary part of the θ -dependent density operator elements $a_{\ell, \theta}^* a_{\ell', \theta}$, and account for quantum state normalization. We recall that frequencies $\{v_\ell\}$ are simply integer multiples of the smallest ('base') frequency $2\pi/2^N$ defined by the register size. Looking at the differences of $\{v_\ell - v_{\ell'}\}_{\ell, \ell'=0}^{2^N-1}$ we observe that the model in Eq. (11) corresponds to Eq. (10) with $\omega \in \Omega = \{0, \pm 1, \pm 2, \dots, \pm(2^N - 1)\} \times 2\pi/2^N$, where multiplicity for each frequency decreases as $2^N - \ell$, $\ell = 0, 1, \dots, 2^N - 1$, and we just need to collect associated coefficients $c_{\omega, \theta}$ for each ω . We thus see that the spectral properties of the phase feature map and associated latent model establish its capacity of exponential size with $(2^N - 1)$ nonzero frequencies, and the same degree (times the base frequency) [71].

Given the analysis above, we draw several conclusions that are highly important for the successful training of quantum generative models. We list them below.

(1) Both DGQM and QCBM have $O(2^N)$ model capacity but have different model expressivity in terms of coefficients $\{c_{\omega, \theta}\}$. As variational unitary circuits have limited depth due to trainability, the performance will widely vary depending on typically accessible model coefficients for the given ansatz [71]. The exponential capacity can then be seen as a problem for certain distributions (see discussion in Ref. [72]), as highly oscillatory terms will lead to overfitting and corrupt derivatives when solving differential equations.

(2) In latent space there is a clear separation between high and low frequency parts of the model, corresponding to qubits with small and large j . This suggests that DGQM can be trained to adjust mostly low frequency components while keeping high frequency components intact, and use the full register for sampling. This is the core of qubitwise training described in the next subsection. We note that such an approach does not hold for QCBMs.

(3) A family of models accessible by DQGM is that of trigonometric polynomials with exponentially many frequencies and constrained variationally controlled coefficients. In

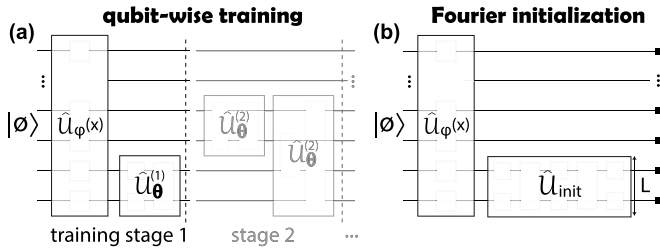


FIG. 2. Frequency-taming techniques. (a) Qubitwise training, where variational circuit is first trained to adjust low frequency part of a model (stage 1). In the second stage we keep $\hat{U}_\theta^{(1)}$ fixed, and train the higher frequency components with $\hat{U}_\theta^{(2)}$, also correlating it with the lower frequency register. This continues until sufficient accuracy. The final optimization run is for the full circuit and register. (b) For the Fourier initialization we first find classical Fourier series for a distribution of interest with $(2^L - 1) \sim \text{poly}(N)$ frequencies, and use \hat{U}_{init} to prepare the corresponding state.

cases where a smooth probability distribution is modeled it may suffice to train only the low-frequency part of the register $L < N$ chosen such that $2^L \sim \text{poly}(N)$. This allows for classical Fourier (cosine/sine) series to be used for probability distribution modeling and/or differential equation solving. The quantum model then requires $O[\text{poly}(N)]$ depth circuit as an instruction for creating the state $\hat{\rho}_\theta$ that matches this series. In this case we can initialize the system close to a predicted solution (performing Fourier series initialization), but still getting sampling advantage for the full register and only using the variational state preparation for inducing further correlations.

(4) The structure of the phase map is quite peculiar—unlike product and tower feature maps [30], where phases of x -dependent rotations are either qubit-independent or have a prefactor of j , the phase feature map has $\sim 2^{-j}$ scaling. Thus, for the same capacity of the phase and product feature maps, the latter has higher expressivity as more qubits and wider variational circuits are used. We address this issue by proposing several feature map ‘frequency-taming’ techniques in the next section.

B. Frequency-taming techniques

In this subsection we describe several strategies that can be used for DGQM training. Specifically, we exploit the knowledge of latent space to perform training in several stages and provide means of regularizing trained generative models.

1. Qubitwise learning

As one of the frequency taming techniques for DQGM training we consider splitting the ansatz into lower and higher frequency parts. We call this qubitwise learning, similarly to the layerwise learning in classical and quantum machine learning [73]. We sketch the procedure in Fig. 2(a), where training is broken into stages. First, the goal is to get the base frequencies right for the model, and qubits $j = N, N - 1, \dots$ are trained. Next, we save quasioptimal angles for the first cycle of optimization, and proceed to include higher frequencies (qubits with smaller j). It is also important to correlate the registers, possibly with a tailored ansatz, and this question

is a matter of future research. Finally, when all quasioptimal angles are found, we perform training for the full register.

2. Fourier initialization

One of the common problems affecting machine learning models is initialization that leads to local minima, and prohibits finding high-quality models. In Ref. [29] we have shown that initialization with low-degree polynomial (truncated Chebyshev series) can vastly reduce number of optimization epochs. Here, we propose to use the structure of the quantum model in Eq. (11), and match coefficients for all frequencies $\omega \in \Omega$ by preparing a suitable quantum state $\hat{U}_{\text{init}}|0\rangle^{\otimes L} = \sum_{\ell=0}^{2^L-1} a_{\ell, \text{init}} |x_\ell\rangle$ [Fig. 2(b)]. Note that the preparation circuit can be exponentially deep in L (see circuit construction in Ref. [74]), but since we only care about $\text{poly}(N)$ frequencies we choose $L \ll N$, suggesting that this is a feasible step for cases where limited expressivity suffices, but fast sampling is needed for dataset augmentation (and specifically relevant for multidimensional distributions).

3. Feature map sparsification

As we noted before, one of the desirable features when working with a feature map of exponential capacity is the possibility to control coefficients for different frequencies. For example, the comparison of serial and product feature maps in Ref. [71] has shown that for the same model capacity the product feature map had better expressivity as already with a layer of rotations one has independent control over multiple coefficients, unlike the serial case. For the phase feature map we are in the situation where feature map rotations are concatenations of base frequency rotations, and no variational control of the model is allowed at that stage—to enable sampling we cannot simply change the feature map as it is an integral part of the measurement circuit. We overcome this issue by proposing the strategy for spreading the features over larger number of qubits, which we name the feature map sparsification strategy.

The idea relies on the fact that we can concatenate two circuits if we use a modified quantum gate teleportation circuit [75]. Note that we have chosen to work in the X Pauli basis for simplicity as the spectrum of the models is the same, and given that $\hat{H}\hat{Z} = \hat{X}\hat{H}$ we simply append an extra layer of Hadamards to the transformation circuit \hat{U}_{T_ϕ} . We show the sparsification workflow in Fig. 3. Concentrating on lowest frequencies, we observe that the second-to-last qubit in the feature map shall be in the $\hat{R}^x(\varphi_{N-1,x})|0\rangle_{N-1}$ state, and $\varphi_{N-1} = 2\varphi_N$. We can prepare the same state by merging two rotations from different qubits. We take a *seed* state as $\hat{R}^x(\varphi_{Nx})|0\rangle_s$ [labeled as s in Fig. 3(a)]. Using a Bell state with an ancilla qubit, we can teleport the state from the seed to the register qubit, such that an additional $R^x(\varphi_{Nx})$ gate is applied. Note that the process can be made deterministic if we add an x -dependent correction circuit. In this case sparsification is performed by the unitary gate \hat{U}_{sp} , and circuit identity in Fig. 3(a) holds.

It is important to stress that we can use sparsification during the training stage, where all qubits (including ancillas and seeds) are trained to match the model—this does not change the frequencies, but increases expressivity. Next, during the sampling stage we then use the trained model, but only sample

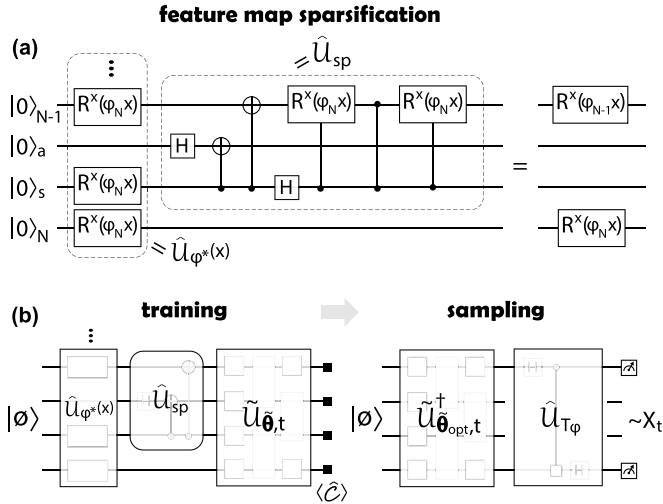


FIG. 3. Feature map sparsification. (a) Low-frequency part of the phase feature map, where the rotation gate from the seed qubit (s) is teleported to the register qubit $N - 1$, which stores the second lowest frequency. Higher-frequency sparsifications can be constructed in the similar way, with varying split in frequencies (degree of sparsification). (b) Training and sampling stages for the sparsified phase map, where the variational circuit acts on all qubits including seeds and ancillas, while during sampling only the N -qubit register is transformed and measured. Again, only lowest frequencies are shown.

qubits from the state register on which the transformation circuit acts.

4. Phase map differentiation

We recall that the DQGM model is built as

$$\tilde{p}_\theta(x) = \text{tr}\{\hat{C}_\theta \hat{U}_\theta \hat{U}_\varphi(x) \hat{\rho}_0 \hat{U}_\varphi^\dagger(x) \hat{U}_\theta^\dagger\}. \quad (12)$$

Our goal is to evaluate $d\tilde{p}_\theta(x)/dx$ analytically (i.e., in a bias-free manner). For this, first observe that

$$\frac{d\hat{U}_\varphi(x)}{dx} = -i\hat{M}_x \hat{U}_\varphi(x), \quad (13)$$

where we introduce the operator $\hat{M}_x := \pi \sum_{j=1}^N \hat{X}_j / 2^j$ as the generator of the phase map (again, we use the X Pauli basis for convenience). We note that it commutes with the map trivially, $[\hat{M}_x, \hat{U}_\varphi(x)] = 0 \forall x$. We recall that $\hat{C}_\theta = \hat{\rho}_0$.

Now we proceed to differentiating the full model, which gives

$$\begin{aligned} \frac{d\tilde{p}_\theta(x)}{dx} &= i \text{tr}\{\hat{\rho}_0 \hat{U}_\theta \hat{U}_\varphi(x) \hat{M}_x \hat{\rho}_0 \hat{U}_\varphi^\dagger(x) \hat{U}_\theta^\dagger\} \\ &\quad - i \text{tr}\{\hat{\rho}_0 \hat{U}_\theta \hat{U}_\varphi(x) \hat{\rho}_0 \hat{M}_x \hat{U}_\varphi^\dagger(x) \hat{U}_\theta^\dagger\}, \end{aligned} \quad (14)$$

where we change the order in which \hat{M}_x acts on $\hat{\rho}_0$. We observe that the corresponding measurement of two overlaps can be combined into the measurement of the expectation value

$$\frac{d\tilde{p}_\theta(x)}{dx} = \text{tr}\{(\delta_1 \hat{C}) \hat{U}_\theta \hat{U}_\varphi(x) \hat{\rho}_0 \hat{U}_\varphi^\dagger(x) \hat{U}_\theta^\dagger\}, \quad (15)$$

where we defined a differential cost operator $\delta_1 \hat{C} := i\hat{M}_x \hat{C}_\theta - i\hat{C}_\theta \hat{M}_x$. Note that the result is valid for both global and local

cost operators. For instance, for the global cost the modified differential cost operator can be rewritten as

$$\delta_1 \hat{C} = \pi \sum_{j=1}^N \frac{1}{2^j} \hat{Y}_j \otimes |\phi\rangle_j \langle \phi|, \quad (16)$$

and the state $|\phi\rangle_j$ simply means that we are in zero for the register of $N - 1$ qubits, apart from the j -th one. We see that we need N evaluations of this expectation. This is an improvement over the $2N$ evaluations for the parameter shift rule. By analyzing the commutators in $\delta \hat{C}$, that correspond to SWAP-like operators, we may possibly do better, and this is a question for future research.

Similarly, we can write a second-order derivative for the quantum probability distribution. For this, we can differentiate the expression in Eq. (16), and observe that $d^2 \tilde{p}_\theta(x)/dx^2$ can be written as an expectation value

$$\frac{d^2 \tilde{p}_\theta(x)}{dx^2} = \text{tr}\{(\delta_2 \hat{C}) \hat{U}_\theta \hat{U}_\varphi(x) \hat{\rho}_0 \hat{U}_\varphi^\dagger(x) \hat{U}_\theta^\dagger\}, \quad (17)$$

where we introduce another Hermitian operator

$$\delta_2 \hat{C} := 2\hat{M}_x \hat{C}_\theta \hat{M}_x - \hat{M}_x \hat{C}_\theta - \hat{C}_\theta \hat{M}_x, \quad (18)$$

which can be decomposed into $O(N^2)$ noncommuting terms and measured separately.

C. Preparing multidimensional correlated distributions

It is unlikely that sampling from a single univariate distribution using a quantum computer gives a computational advantage over using a classical computer. In the end, for most practical cases we can use—for example—a finite-degree polynomial approximation. This is commonly used in financial analysis. However, when working with multivariate (multidimensional) distributions, sampling becomes complicated. This prompts us to consider problems comprising of a D -dimensional vector of stochastic variables $\mathbf{X} = (X_1, X_2, \dots, X_D)$. The underlying probability distribution corresponds to $p(\mathbf{x})$ with $\mathbf{x} = (x_1, x_2, \dots, x_D)$, and often it is convenient to work with a multivariate cumulative distribution function $F(\mathbf{x})$. If the distributions are not correlated, then we can do inverse sampling assuming that the multivariate CDF factorizes into a product of marginal distributions, $F_{\text{simple}}(\mathbf{x}) = F_1(x_1) \cdot F_2(x_2) \dots F_D(x_D)$, and the same is true for the probability density function. This means, even though we consider multivariate distributions, the simulation can be parallelized efficiently following the univariate case. However, for correlated variables this decoupling procedure is not valid. Classical simulation of multivariate distributions and corresponding generative modeling is generally difficult. Potential approaches include delayed rejection adaptive Metropolis algorithm, and the state-of-the-art protocols based on a tensor train decomposition [76]. In general, they assume truncation of correlations, and the full generative modeling requires including fine structure, at large computational cost.

Another notable approach for dealing with multivariate distributions corresponds to quantum generative adversarial networks, specifically in cases where classical discriminator networks are used [57,77]. QGAN-based generators are known to excel when working with complex multidimensional

distributions, and this has been demonstrated to translate well into the quantum regime when generators are based on QCBM-type circuits [78]. We consider the multivariate QGAN a suitable choice when no priors on distributions (i.e., physics-informed processes) are available.

Let us now proceed to describe an approach specific to DQGM models that can be used for time-series based on underlying SDEs and differential constraints. A way for including correlations between stochastic variables can be provided by quantum hardware, as quantum systems are good at correlating subsystems. Recently, generative modeling was shown to benefit from correlation, and specifically entanglement [56]. One way to think about it is simply consider a joint register for the vector of variables \mathbf{x} . However, in this case we are left with a QCBM-type problem of enlarged size, and training for large D can become prohibitive. A more subtle way corresponds to including correlations by encoding copulas into quantum hardware, as recently proposed in Ref. [57].

The concept of copula was developed to yield multivariate sampling by correlating latent variables, while keeping the sampling procedure individual to each variable. Imagine a bivariate distribution such that two stochastic variables X_1 and X_2 are distributed normally, but are in fact correlated. The correlation for normal distributions can be accounted using a covariance matrix, which grows with the dimension D . Thus, accounting for correlations again becomes challenging for generic D -dimensional distributions. However, this problem can be resolved by introducing a copula—a function that links marginal distributions of different variables [79]. Copulas absorb correlations between variables while being agnostic to the type of marginal distribution. Specifically, following Sklar’s theorem we write a copula $C[\mathbf{v}]$ acting on some vector \mathbf{v} as a function

$$F(\mathbf{x}) = C[F_1(x_1), F_2(x_2), \dots, F_D(x_D)], \quad (19)$$

which links marginals into a full multivariate CDF. Similarly, a copula density function $c[\mathbf{z}]$ for the latent variable vector \mathbf{z} is defined as

$$c[\mathbf{x}] = c[F_1(x_1), \dots, F_D(x_D)]p_1(x_1) \cdot \dots \cdot p_D(x_D). \quad (20)$$

A useful property of copulas is that by generating a vector of samples from the copula as $\mathbf{Z} = (Z_1, Z_2, \dots, Z_D) \sim c$, we can transform them into samples of the original multivariate distribution as [79]

$$\mathbf{X} = (Q_1(Z_1), Q_2(Z_2), \dots, Q_D(Z_D)), \quad (21)$$

where $Q_j(Z_j)$ are marginal quantile functions (inverted CDFs) for distribution of j th stochastic variable. Here, we stress that copula produces correlations at the level of latent variables, as used in the inverse sampling [29]. It represents a modified PDF that deviates from a uniform multivariate distribution, and thus correlates the outcomes for multivariate PDF sampling.

Since the copulas capture correlations only, while having flat marginals, they can be modeled by entangled states [57]. Namely, the correlations can be introduced using a quantum circuit of finite depth that is applied *prior* to separate variational registers (see Fig. 4). Yet, when we link D registers, even for tractable N -wide individual distributions, we are left with $D \times N$ qubits that are maximally entangled, in the logical

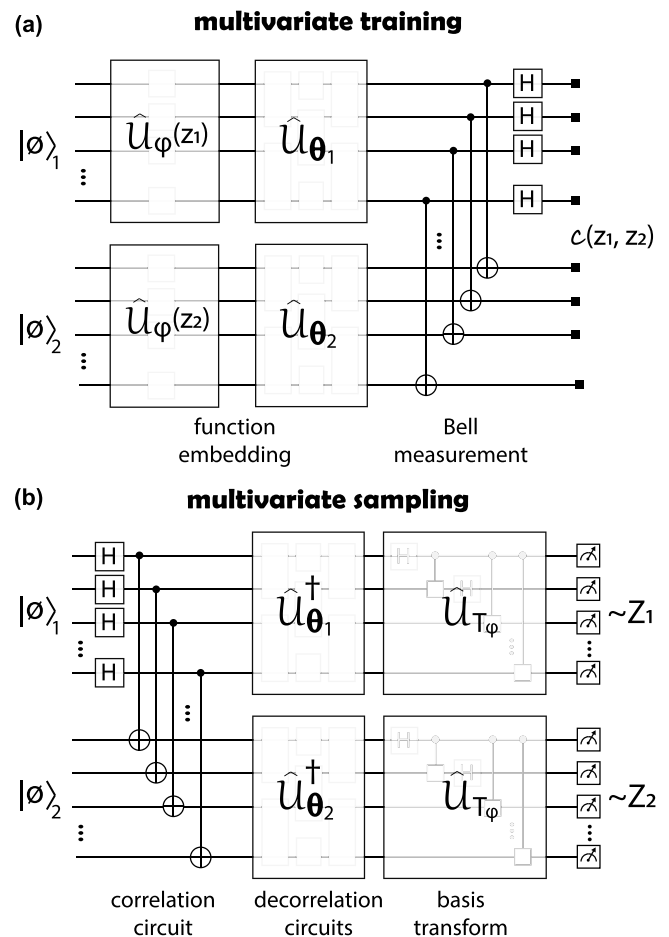


FIG. 4. Multivariate quantum generative models based on copulas. (a) A model is trained to represent a copula dependence for latent variables, where the correlation between registers is included as series of Bell measurements. (b) The trained model is sampled in the bit basis starting from the cluster state that is transformed by variational circuits.

sense. As we form a cluster state, this requires the bond dimension to go up, preventing efficient classical simulation. This is the setting in which we expect to get an advantage in quantum generative modeling.

We propose to build a quantum generative model for copulas, expressing it as a function of latent variables encoded using the phase feature map. The corresponding circuits for quantum copula modeling are shown in Figs. 4(a) and 4(b). First, the copula PDF is constructed as a function of variables \mathbf{z} using the feature map encoding. We note that both DQGM and generalized QCBM models can be built. In the former case one needs to think in terms of frequencies, and in the latter case one shall think in terms of bit strings. The model is then constructed by first applying variational circuits on separate registers, then followed by the Bell circuit measurement and expectation of the cost operator \hat{C}_ϕ (global or local) [see Fig. 4(a)]. Intriguingly, this setting is similar to learning from data that has shown a great promise recently [80], and uncovering the relation between two subjects is an interesting avenue for the future research. Once we trained the model for copula, we can revert the circuit, and read out samples in the transformed basis for DQGM [Fig. 4(b)]. Note that

the probability density function remains the same. For the generalized QCBM, we note that \hat{U}_{T_ϕ} is a part of training, while being absent in the sampling stage.

We highlight that while building a quantum generative model for copulas, one can build powerful intuition about processes in the system. First, we observe that by generating a cluster state and using identity operators instead of variational circuits one enforces maximally correlated samples of $c(z_1, z_2)$. This in turn leads to strong correlation for samples $\mathbf{X} \sim p(\mathbf{x})$. However, by performing local operations on registers of separate variables one can effectively decorrelate their samples in the copula space, and thus in the space of multivariate PDF samples. We elaborate on this point in the Results section considering an example based on a Gaussian copula for bivariate distributions [81].

Furthermore, the importance of representing a copula as a differentiable quantum model comes from the fact that for many stochastic processes (for instance, in financial modeling) certain copulas are shown to perform well, and represent an excellent starting point [79]. Going beyond learning from data, one can use knowledge of differential constraints when learning copulas. This creates inherent regularization and helps capturing properties specific to the process. For instance, the system of Fokker-Planck equations formulated for a copula PDF, and used as a differential constraint, may offer an edge when training copula circuits [82].

III. RESULTS

To test the proposed protocols, we conduct several numerical experiments. The goal is to showcase the performance of the DQGM approach in various scenarios, spanning from explicit learning of distributions to solving stochastic differential equations, propagating them in time, and addressing multivariate problem. As an underlying model for this we choose the Ornstein-Uhlenbeck process [28]. Being a starting point for the Hull-White and Vasicek models, Ornstein-Uhlenbeck SDE helps with, amongst others, modeling currency exchange rates [24]. First, we test the approach on learning a static distribution. Second, we introduce differential constraints and solve the steady-state FPE for OU. Third, we evolve the learnt solution in time, specifically solving the time-dependent FPE for OU using the implicit time embedding. Finally, we present results for multivariate sampling with quantum copula models.

A. Learning generative models

We start with representing a probability density function (PDF) by DQGM circuits, with consequent sampling. Here, the goal is to understand the expressivity of the proposed generative models and their capability of representing typical PDFs. Additionally, we aim to compare it to the generalized QCBM architecture and highlight the differences in training. We choose the target distribution that corresponds to a normal process (Ornstein-Uhlenbeck being one example). The corresponding PDF reads

$$p_{\text{target}}(x) = \frac{1}{\sqrt{2\pi\sigma_0^2}} \exp\left[-\frac{(x - \mu_0)^2}{2\sigma_0^2}\right], \quad (22)$$

where μ_0 is a mean and σ_0^2 is a variance. We note that to be able to load a PDF in a quantum register suitable parameter scale should be chosen. Namely, μ_0 and σ_0 are chosen such that the probability can be potentially stored in a register of N qubits with $x \in [0, 2^N - 1)$ and $0 \leq p(x) \leq 1$. We choose the mean square error (MSE) as a loss, which is normalized by the number of samples at which distributions are compared. As a testing ansatz for simplicity we use a hardware efficient ansatz (HEA) [83] with alternating $SU(2)$ rotations and CNOT-based entangling layers. Specifically, we compose a variational circuit of d layers and width w . Here, $d = 0$ corresponds to single $SU(2)$ layer (for instance, decomposed into $X - Z - X$ parametrized rotations), followed by d repetitions of CNOTs on odd/even sublattices and $SU(2)$ layers. The parameter w defines on how many qubits the variational ansatz acts, starting from the bottom one (lowest frequency). For instance, $w = 3$ for $N = 6$ register means we only use qubits $j = 4, 5, 6$, and act with an identity on the rest. Variation is performed using gradient-based Adam optimizer, and we use Julia's Yao package as a simulator [84].

We start by considering a target distribution with $N = 6$ qubits. We set the mean to $\mu_0 = 32$ and the standard deviation of $\sigma_0 = 8$. The training grid is set up to include all integer points of x , and we use a thousand of epochs. The training is performed for varying depth and width. We test the performance of both DQGM and generalized QCBM for modeling the target as well as providing samples. As a metric, we plot the quality of the solution, represented by the MSE loss evaluated for twenty times more points (referred as a generalized grid). The results are shown in Fig. 5. In Fig. 5(a) we show the quality of solution for DQGM at the end of training. We observe that at full width training the model contains exponential number of frequencies, limiting the performance due to large 'out-of-sample' error. At the same time, with smaller width we can capture the target distribution using lower frequency components, and reach high-quality solutions. While the performance is likely to be model dependent, we observe that the optimal solution requires choosing a suitable combination of w and d which we discuss further later. As an example of trained PDF we pick $d = 4$ and highest-performing width of $w = 3$. This can be seen as a simplest instance of qubitwise learning, and generally highlight the relevance of frequency-taming. The trained DQGM closely follows the target model at all points [see Fig. 5(b)]. We then apply the basis transformation and sample our model with the extended register of $M = 10$ qubits. The histogram is shown in Fig. 5(c), where 10^7 shots are used, and we normalize bins over the total number of samples.

Next, we consider the performance of generalized QCBM for the same problem. The results for d and w scanning are depicted in Fig. 5(d). As encoding assumes transformations on bitstrings, smaller w circuits do not perform well, and $w = N$ is required, as expected. We note that the presence of high frequencies in the model and absence of regularization that limits high frequency components generally impacts the QCBM's performance. The instance with the best quality is shown in Fig. 5(e). While overall the shape represents the distribution well, high-frequency components impact the model quality as it does not generalize. The impact on solving differential equations based on such a model will be tremendous. This

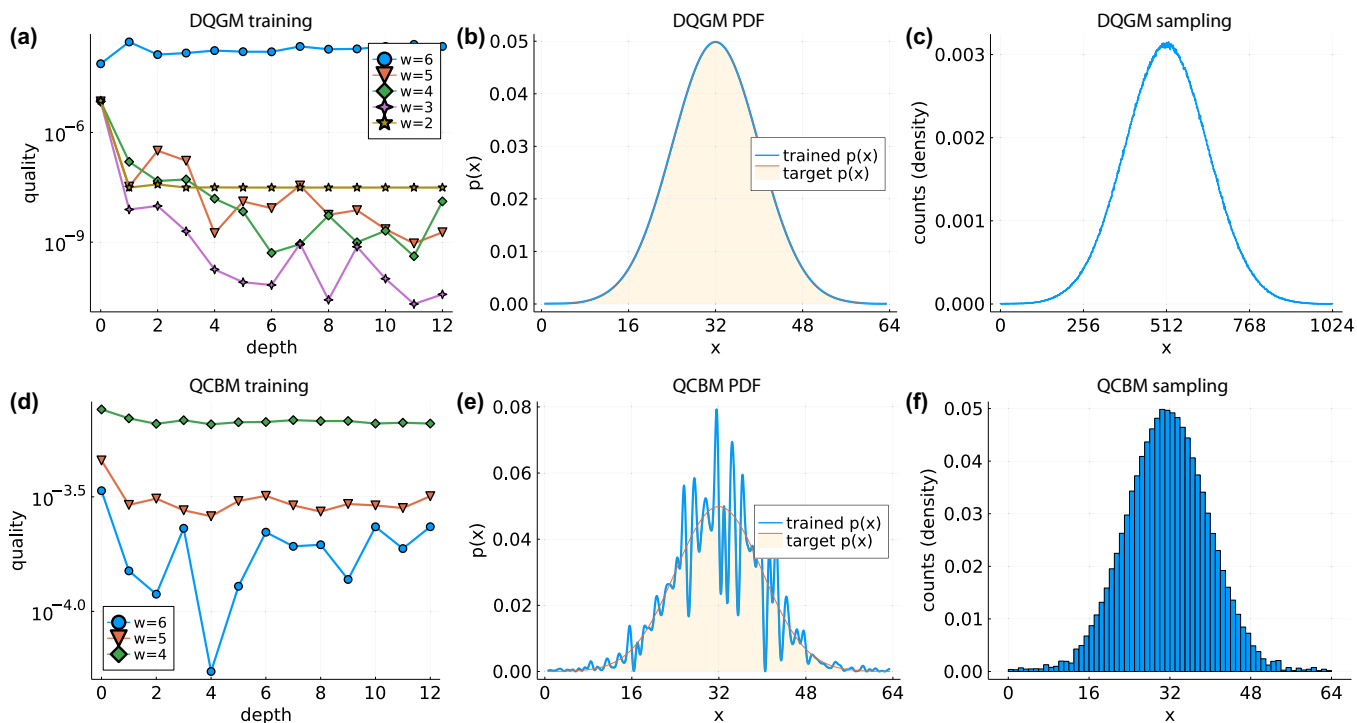


FIG. 5. DQGM and QCBM comparison. (a) MSE loss for DQGM trained at different depths and widths, showing quality of solution on the generalized grid. This corresponds to the quality metric, where smaller numbers (deviation) means higher quality. (b) PDF from the DQGM training at $d = 4$ and $w = 3$. (c) Sampled probability distribution from transformed DQGM using $N = 10$ qubits and 10^7 samples at the readout. (d) Quality metric based on the MSE loss for the generalized QCBM trained at different depth and width. (e) Best model for QCBM shown for $d = 4$ and $w = N$. (f) QCBM samples from $N = 6$ qubits and 10^6 shots.

can traced directly to the exponential capacity of the phase feature map, and the absence of simple frequency-taming. One option for regularization here is including more points during training, but this comes at the price of training on dense grids. Finally, we show the sampling from generalized QCBM in Fig. 5(f). The histogram qualitatively matches with the target, as requested by optimization loss.

Following the use of the variational approach, we have also implemented the initialization procedure. In this case the target distribution is expanded in cosine series for 4 qubits, such that the coefficients of the preparation state are known. Using a $\mathcal{SO}(2^4)$ circuit that can create an arbitrary real-amplitude state, we efficiently utilize all frequencies. The resulting PDF is shown in Fig. 6. We note that initialization may be required in cases where we want to off-load part of job from the variational procedure.

B. Solving stationary Fokker-Planck equations

We proceed to introduce differential constraints, where together with learning from data by minimizing $\mathcal{L}_\theta^{\text{data}}$, we wish to minimize $\mathcal{L}_\theta^{\text{diff}}$ coming from the FPE differential operator. Here the goal is understanding how the addition of physics-informed (or financial process-informed) terms helps to guide the DQGM training, leading to meaningful models with correct sensitivities. We note that other approaches like QCBM cannot include such loss terms, highlighting the increased capabilities of the DQGM approach. While the data-based learning does not require knowing the model parameters per se, the SDE/PDE/ODE learning does depend on the model

parameters introduced by the drift and diffusion terms. We again choose the Ornstein-Uhlenbeck process as it lies at the core of many financial models. SDE of the OU process corresponds to static drift and diffusion terms, and reads

$$dX_t = -v(X_t - \mu)dt + \sigma dW_t, \tag{23}$$

where μ , σ , and v are model parameters, which can be discovered while learning from data. Using Eq. (9) we can see

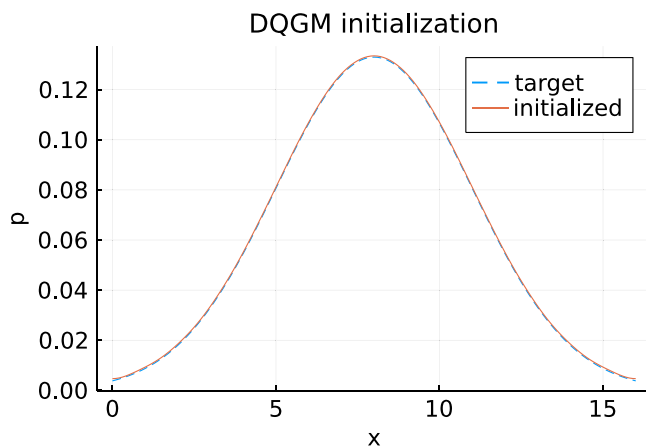


FIG. 6. Fourier initialization of DQGM. We use a cosine expansion and initialize the circuit for $L = N$, reaching high-quality solutions and exploiting the full spectrum of $N = 4$ DQGM.

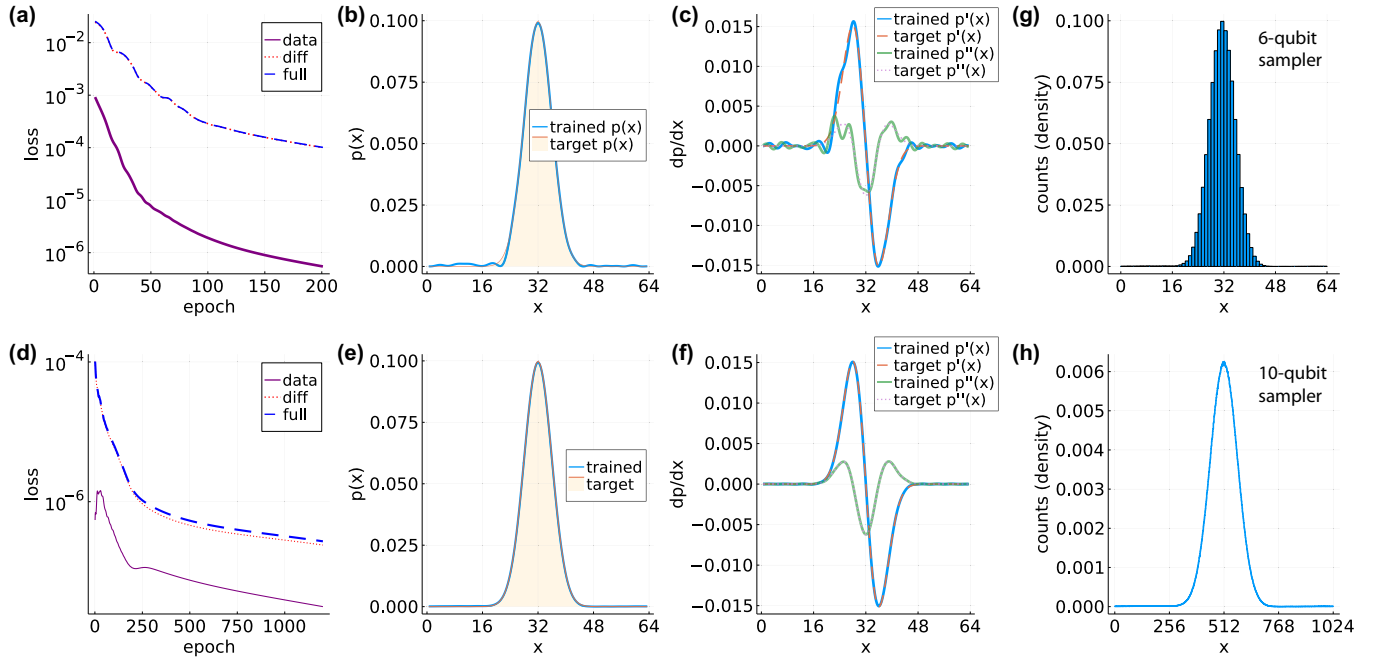


FIG. 7. DQGM trained to sample from the Ornstein-Uhlenbeck process by matching the steady state of FPE. (a) History of the data training, showing the data loss which is used for optimization. Differential loss (labeled as “diff”) and the full weighted loss are plotted for comparison. (b) Probability distribution $p(x)$ from the data-trained DQGM, where small number of epochs is used. (c) Derivatives of the model trained on data. (d) History of DQGM training with differential constraints (stationary FPE), where the full weighted loss is used for optimization, and the other two loss functions are plotted for comparison. (e) The probability distribution function from DQGM trained on the full loss. (f) Derivatives of the generative model based on FPE constraints. (g) Normalized sampling histogram for $N = 6$ DQGM trained using FPE differential constraints, where 10^6 shots are measured. (h) Normalized sampling histogram from an extended 10-qubit register.

that at the steady-state FPE for OU corresponds to

$$\nu p(x, t_s) + \nu(x - \mu) \frac{d}{dx} p(x, t_s) + \frac{\sigma^2}{2} \frac{d^2}{dx^2} p(x, t_s) = 0. \quad (24)$$

Notably, when starting from some initial mean, we arrive to μ as a new mean in the steady state (at the rate of ν), and the variance $\sigma^2/2\nu$. It is convenient to set $\nu = 1$, assuming that time is in units of ν^{-1} .

In the following we assume that OU reached the steady state, and learn the corresponding distribution from the differential constraints. The workflow is as follows. First, we choose SDE/FPE parameters as $\mu_0 = 32$ and the variance of $\sigma_0^2 = 32$. The quantum model is set up with $N = 6$ qubits, $d = 4$ and $w = 3$ as suggested by previously performed depth scanning. We set up three different loss functions to track the performance during training. The first two correspond to the data loss $\mathcal{L}_{\theta, t_s}^{\text{data}}$ and the differential loss for static FPE, $\mathcal{L}_{\theta, t_s}^{\text{diff}}$, as described in the second section. The third loss, which we call the full loss, is then taken as a weighted average of data and differential contribution, $\mathcal{L}_{\theta, t_s}^{\text{full}} = \mathcal{L}_{\theta, t_s}^{\text{data}} + \eta \mathcal{L}_{\theta, t_s}^{\text{diff}}$, where coefficient η controls the weight of FPE constrained (this is generally needed as the two may be imbalanced even when normalized over the grid). We perform DQGM training in two stages. At first, our goal is learning the initial condition of FPE, where the gradient descent is performed on $\mathcal{L}_{\theta, t_s}^{\text{data}}$. We deliberately choose a coarser grid with 32 points of x and 200 epochs, simulating imperfect training conditions (i.e., when knowledge of probability distribution is not available, and data is noisy). The results are shown in Figs. 7(a)–7(c). Looking at the history, the training loss goes down promptly, yet we

observe a large separation between the data and diff loss contributions [Fig. 7(a)]. In Fig. 7(b) we show the corresponding PDF which captures the data well. Yet when plotting derivatives of the target model and DQGM in Fig. 7(c) significant deviations are visible. The latter can impact predictions when considering out-of-sample examples. In the second stage we turn on the differential loss, and the full loss with equal contributions ($\eta = 1$). We use angles from the data training. Smaller learning rates are used to avoid jumping far from the previously found valley in a landscape, and we simulate 1200 epochs. The full loss goes down together to much lower values [Fig. 7(d)]. This translates into a high-quality PDF [Fig. 7(e)]. But most importantly, the presence of differential constraints provided high-quality derivatives plotted in Fig. 7(f). This paves the road to *training models*, and not just learning from data, especially in cases where large datasets are not available or cannot be loaded efficiently. We complete static FPE learning by sampling from optimal DQGM, based on the full loss. The originally trained and extended 6- and 10-qubit sampling shown in Figs. 7(g) and 7(h), showcases improvements offered by including the knowledge about the model and underlying SDE/PDE.

C. Solving time-dependent Fokker-Planck equations

Once the initial state is learnt and differential constraints are accounted for, we may ask an additional question: Can we predict the trajectories of the stochastic process that lead to the steady state? To answer the question, let us first solve the problem using the conventional Euler-Maruyama technique

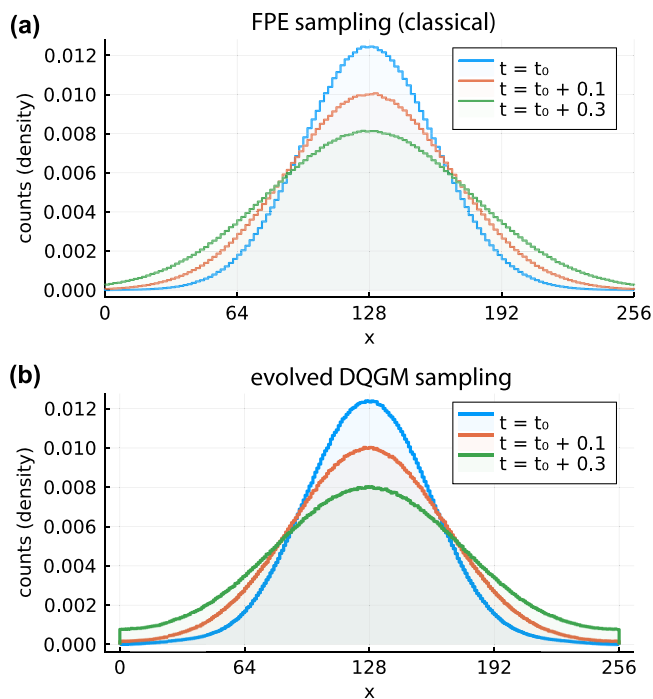


FIG. 8. Time-dependent SDE sampling. (a) Samples from classically evolved distribution at different time points (t_0 , $t_0 + 0.1$, $t_0 + 0.3$). (b) Samples from time-evolved DQGM at the same three times obtained by evolving circuit parameters with the implicit time embedding.

[85]. In this subsection we aim to show that DQGM is able to address problems which are described by distributions evolving in time. While similar evolutionary approaches are also applicable to implicit QCBM-type models, the advantage of using DQGM is in preparing high-quality distributions and their derivatives, which enable time-propagation (for trained PDFs with visible oscillations the propagation becomes unstable).

We set up an SDE solver for the OU process with increasing variance. For simplicity, we consider a process without mean reversion, setting $\mu = 32$, and a variance of $\sigma = 512$ as SDE/FPE parameters. We start from the Dirac delta function distribution at zero time, and learn the PDF at $t_0 = 0.144$ (in the units of inverse κ). At this point the distribution matches the variance of 64, and continues to grow thereafter. The results from classical SDE sampling are shown in Fig. 8(a) for three different times being t_0 , $t_0 + 0.1$, and $t_0 + 0.3$, chosen such that changes are significant. Next, we perform time-evolved simulation with the DQGM. We express the solution as DQGM at t_0 with a $w = 2$ and $d = 1$ circuit that performed well before, while choosing a variational circuit structure with real-amplitude states (layers of parametrized Y rotations and CZ gates). The training follows data-based loss and 500 epochs. Then, we assume the implicit time embedding, and update parameters of the model θ_{opt, t_0} from the initial ones by evaluating the FPE operator and Jacobians. We use a simple Euler's scheme with $\Delta t = 0.001$ and three hundred steps. Note that this may lead to instability for longer propagation times, where Runge-Kutta and stencil-point methods are preferred. The histograms for time-evolved DQGM are shown

in Fig. 8(b), where 10^7 samples are used. We observe good agreement with classical sampling, and note that having a smooth model the sampling can further be extended to larger register sizes. We also note that explicit encoding may be beneficial for situations where we need to generalize in time. This will be a question for future research on the topic.

D. Sampling from bivariate normal distributions

Next, we study a pedagogical example of sampling from a multivariate distribution. Here the goal is to understand how DQGM can be taken further and consider correlated distributions of many random variables. Indeed, univariate distributions of various type can be most often prepared and sampled classically. But in the case of multivariate nature PDFs are difficult to invert. We note that the power of DQGM approach is in ability to work explicit forms of distributions, adding priors that remain inaccessible to implicit QCBM modeling.

We consider a bivariate normal distribution $p(x_1, x_2)$. This type of distribution can be fully characterized by its mean values for each stochastic variable $\mu_{1,2}$, their respective standard deviations $\sigma_{1,2}$, and importantly the correlation parameter ρ_{12} . Let us first analyze different examples using known classical procedures of inverse sampling which accounts for the covariance matrix. In Figs. 9(a)–9(c) we show three examples for classical bivariate sampling. The first example concerns highly correlated samples (X_1, X_2) with $\rho_{12} = 0.999$, each normally distributed with $\mu_{1,2} = 0.5$ and $\sigma_{1,2} = 0.1$ [Fig. 9(a)]. One can think of financial processes with similar correlation at highly regulated markets or, for instance, looking at EUR-DKK currency pair. Next, as a reference we show sampling from uncorrelated distribution with $\rho_{12} = 0$ [Fig. 9(b)], which is equivalent to separate inverse sampling of X_1 and X_2 , plotted together. The third example in Fig. 9(c) concerns a negative correlation value of $\rho_{12} = -0.5$. This example is relevant in cases when significant but not absolute dependence of two processes is present.

We continue the analysis in the quantum domain using copula as a tool. First, we note that for multivariate normal processes the Gaussian copula PDF $c(\mathbf{z})$ can be expressed as

$$c(\mathbf{z}) = \frac{1}{\sqrt{1 - \rho_{12}^2}} \exp \left\{ [2\rho_{12}Q_1(z_1)Q_2(z_2) - \rho_{12}^2(Q_1^2(z_1) + Q_2^2(z_2))] / 2(1 - \rho_{12}^2) \right\}, \quad (25)$$

where $Q_j(z_j)$ are standard normal quantile functions for variables $j = 1, 2$ expressed as (shifted) inverse error functions parametrized by (μ_j, σ_j) [86]. Now, let us look at the limiting cases. For $\rho_{12} = 0$ the copula PDF becomes the uniform distribution for both variables. In the limit of $\rho_{12} \rightarrow 1$ we get maximal correlations, such that it is given by the Dirac δ function, $c(z_1, z_2) \sim \delta(z_1 - z_2)$. For nonzero ρ_{12} the structure is introduced, leading to preference of some samples over others. Using the described intuition from Gaussian copula, we note that perfect correlation of $\rho_{12} \rightarrow 1$ is readily modeled by a cluster state circuit with variational circuits being identities (here, it is easier to use the generalized QCBM picture for gaining the intuition). Once the copula circuit is set up, we

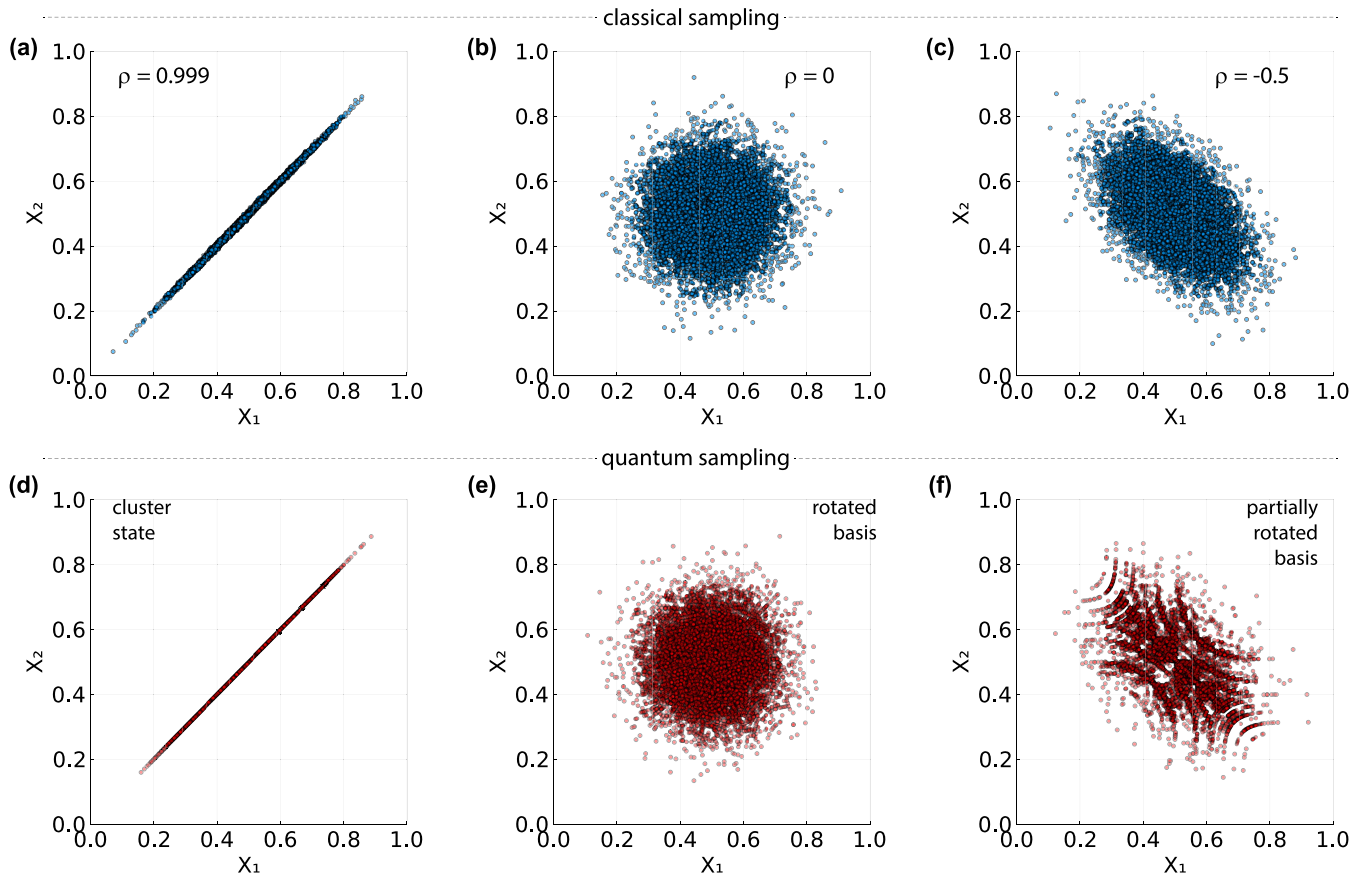


FIG. 9. Classical and quantum multivariate sampling with normal copulas. (a, b, c) Scatter plots for classical sampling of random variables X_1 and X_2 from the bivariate normal distribution. The probability density functions are centered at 0.5, standard deviations are 0.1, and correlation between variables is $\rho_{12} = \{0.999, 0.0, -0.5\}$ for panels (a), (b), and (c), respectively. 10^4 samples are shown. (d) Scatter plot for quantum generative modeling from the maximally logically entangled copula transformed into normal samples and mimicking $\rho_{12} \rightarrow 1$ case. Here and below $N = 12$ qubits are used for the full register, sample values are normalized to be in $[0,1]$ dividing by $2^{N/2}$, and 10^4 are plotted. (e) Sampling from uncorrelated registers, where copula circuit has uncorrelated bases for the two registers. (f) Partially correlated copula transformed into bivariate samples that mimics negative $\rho_{12} = -0.5$ correlation.

perform mapping $Z_{1,2} \rightarrow X_{1,2}$ as described in Eq. (21), and present scatter plots for (X_1, X_2) .

The resulting samples in the multivariate data space are shown in Fig. 9(d), resembling the highly correlated case discussed before. Next, the decorrelation circuit can be set up such that the measurement for Z_1 and Z_2 are performed in different bases, for instance, acting with Hadamards on the first register. The corresponding sampling is shown in Fig. 9(e), mimicking $\rho_{12} = 0$ case. Finally, by employing single qubit rotations on the first register in X and Y basis the partial correlation can be reproduced [see Fig. 9(f)].

We note that the present study shows only the first steps in understanding multidimensional correlated sampling from quantum circuits. However, using the developed tools and combining with knowledge of stochastic processes may improve this understanding ever further.

E. Scalability

Let us now describe the scalability of the DQGM approach as an explicit-implicit modeling tool. Here the goal is highlight modalities in which the approach can be used in

generative modeling, and also when mapping implicit models to explicit (feature map-based models) for the readout. For this, we first discuss the closest relative being the implicit quantum circuit Born machine. The scalability of QCBM with the number of qubits N depends on the ability to work with 2^N state amplitudes (exponentially increasing expressivity), but this implies decreasing trainability [87]. In our work we connect this to generalized QCBM, where models show the presence of exponentially many frequencies. As DQGM scales we also expect the expressivity to grow exponentially, but this can be: (1) limited by keeping only low frequencies in training; (2) applying the frequency taming techniques outlined in Sec. II B. Concerning the Fourier initialization, the cost can be exponential in the subsystem size N_{init} for generic distributions (e.g., using Grover-Rudolph protocol), but due to high expressivity it is sufficient to keep N_{init} small. Alternatively, one can use efficient state loading for smooth functions developed in Ref. [62]. Crucially, there is no direct analogy of initialization for QCBM (unless designing a highly specific sparse circuit for that matter).

For the number of iterations, we note that both QCBM and DQGM are based on nonconvex optimization method. In

general, finding optimal angles for these models is NP-hard [88], and there is no provable guarantees on the number of iterations to converge to the global optimum (unlike for convex optimization schemes). However, the statement concerns most of machine learning protocols including deep neural networks, where excellent performance is achieved through a range of heuristics. This means that with a suitable model parametrization one can expect converging DQGM/QCBM to a required quality, but this depends on the model and cannot be easily bounded.

For the depth of the QCBM and DGQM circuits, it largely depends on the ansatz chosen when building the model. However, DQGM has an advantage of frequency-band selection or separation when training, unlike QCBM that requires uniform layers. Otherwise, both typically use a hardware efficient ansatz (HEA) of depth d scaling linearly $O(N)$ in the system size N , such that one can generate entangled states that support nontrivial models. Other options include restricted ansatzes with shared parameters (QAOA) or embedded symmetries.

Moreover, the depth of the required circuits has to account for the trainability as well. In fact even $O[\log(N)]$ -deep HEA can be untrainable for the QCBM-type training due to the global loss function, as highlighted in Ref. [89]. On the contrary, DQGM has the option of training with the local loss, avoiding the corresponding concentration. As we go to larger d , the trainability drops, and to restore this once may need to use overparameterization [90] (in principle, requiring an exponential depth scaling). However, there is also a “goldilock” zone [91] where HEA with limited depth is still trainable and sufficiently expressive, which we can utilize for DQGM.

When considering the number of trained qubits w and the corresponding circuit depth d , we balance the model expressivity and trainability, as well as quantum resources required for the algorithm. There is a set of constraints to fit, as usual when working with variational circuits. To be more specific, we have added the discussion of d and w choices for HEA that can effectively combine high-expressivity with sufficient trainability, as presented in Ref. [91].

Specific to our protocol and generative modeling, we outline a set of principles summarized as follows. The choice with small w limits number of basis functions and therefore expressivity (also referred as model capacity). As w grows, more basis functions become available and expressivity increases as $\sim 2^w$. The depth d dictates the accessibility of different basis functions, and therefore d is required to scale with w at least linearly. However, the trainability and the onset of barren plateaus sets a restriction on either keeping d small (trainable) but tailoring the ansatz structure such that it is problem specific. Another options is overparametrizing an ansatz [90], which for small w is possible in $O[\text{poly}(w)]$ or $O(2^w)$ depth, depending on the ansatz.

Finally, we note one distinct advantage that is specific to DQGM. This corresponds to the targeted readout of functions represented as $f_{\theta^*}(x) = |\langle x | \psi_{\theta^*} \rangle|^2$. In this case, instead of sampling all possible values and selecting the outcomes for the chosen point x , the DQGM allows evaluating the function at x via the phase feature map. This simplifies the readout when models require extracting information only for a limited set of points [92].

IV. CONCLUSIONS

We developed protocols for efficiently training differentiable quantum generative models, which we refer to as DQGM. Separating training and sampling stages, we train circuits in the latent space as a feature map encoded differentiable circuit, and sample the optimized circuit with additional (fixed) basis transformation. On a technical side, we introduced the phase feature map, analyzed its properties, and developed frequency-taming techniques that include qubit-wise training and feature map sparsification. For numerical simulations, we benchmark the approach against QCBM and show how samples from propagated stochastic differential equations can be accessed by solving a Fokker-Planck equation on a quantum computer. We highlight that the proposed approach is especially suitable for generative modeling tasks which result from stochastic differential equations, as it allows implementing differential constraints into the generative workflow. Our approach also sheds light on a path to multidimensional generative modeling based on copulas, where qubit registers are explicitly correlated via a (fixed) entangling layer. In this case quantum computers can offer advantage as efficient samplers, which perform complex inverse transform sampling enabled by fundamental laws of quantum mechanics. At the same time we note that other approaches are available when working with multivariate distributions with SDE knowledge, where multivariate QGAN is a notable example.

We note that our approach and associated findings only started to uncover the relation between models built in latent spaces and their connection to equivalent models that can be sampled efficiently. So far we have employed the phase feature map [Eq. (1)] which results into the Fourier basis. Other options become available when \hat{U}_ϕ is modified, and even becomes a general CPTP map or a linear combination of unitaries. For instance, this enables working in the exponentially large basis of orthogonal Chebyshev polynomials [93] or Hartley kernels [94]. As the choice of basis is often crucial to solving differential equations [92], we believe that the development of new latent spaces paired with DQGM will push further the boundaries of quantum machine learning and generative modeling.

ACKNOWLEDGMENTS

The authors thank Pasqal for the support. O.K. acknowledges the funding from UK EPSRC (Award No. EP/Y005090/1).

APPENDIX: ADDITIONAL SIMULATIONS

Throughout this work the results have been centered around processes following normal-type distributions, as Gaussians can be expressed by quantum circuits with smaller registers. To highlight that our algorithm is not restricted to such probability distributions we carry out some further simulations on two more distributions. Specifically, below we show the results for log-normal and Pareto distributions.

A log-normal distribution has probability density function of

$$p_{\text{target}}^{\text{logn}}(x) = \frac{1}{x\sqrt{2\pi\sigma_0^2}} \exp\left[-\frac{(\ln(x) - \mu_0)^2}{2\sigma_0^2}\right], \quad (\text{A1})$$

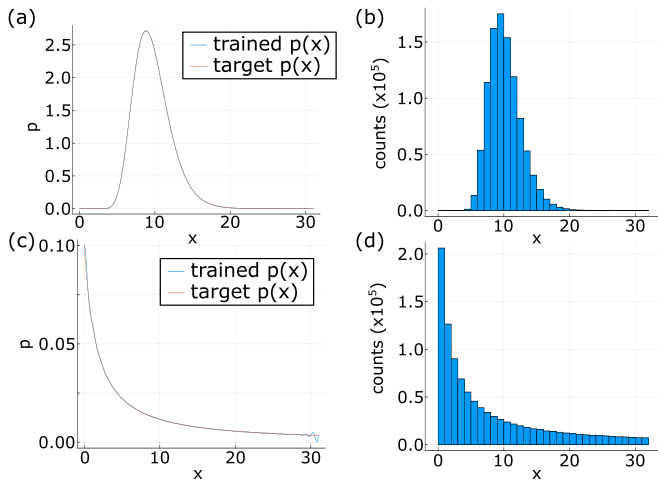


FIG. 10. Simulation of regression for log-normal and Pareto distributions. Both examples are shown for five-qubit registers and for 63 training points uniformly distributed between 0 and $2^N - 1$. (a), (b) Results for log-normal distribution with $\mu = -0.5$, $\sigma = 0.25$. (c), (d) Result for Pareto distribution with $\alpha = 0.2$, $x_m = 2$. (a), (c) Plots of target density function and result from training as function of x . (b), (d) Histogram of 10^6 samples from result of training resulting in panels (a) and (b), respectively.

where μ_0 and σ_0 are parameters that define its shape and moments. The log-normal distribution is similar to the normal distribution though differs by the factor of $1/x$ and additional $\ln(\cdot)$ scaling of the variable in the power of exponent. The distribution of a product of a set of positive independent random variables is log-normal. This distribution appears in biology, chemistry, medicine, social sciences and other areas of science. Some particular appearances are concentration of rare elements in minerals, incubation period of diseases, and the distribution of income (exempting top 1%).

In our simulation we perform regression on this distribution with $\mu_0 = -0.5$, $\sigma_0 = 0.25$ as well as x rescaled uniformly by a factor of 2^{N-1} such that the same scale of distribution is considered as qubit number N increases. The simulation is implemented in the same way as before. We use $N = 5$ qubits and 63 training points uniformly distributed between 0 and $2^N - 1$. The results of this are shown in Figs. 10(a) and 10(b). As can be seen the target and trained result are near identical. Notably, DQGM-based workflow allows representing asymmetric distribution without major deviations.

Additionally we consider the Pareto distribution. This is further distinct from the distributions considered so far with density function

$$p_{\text{target}}^{\text{pareto}}(x) = \frac{\alpha x_m^\alpha}{x^{\alpha+1}}. \quad (\text{A2})$$

It is linked to what is commonly known as the 80–20 rule which states how for certain processes 80% of outcomes are due to 20% of causes. This distribution appears in a variety of situations such as the rate of hard disk drive error rates, the size of human settlements and the income of the top 1% of earners.

Our simulation is implemented in the same way as previous normal and log-normal examples, assuming $\alpha = 0.2$, $x_m = 2$. The results of this are shown in Figs. 10(c) and 10(d). As can be seen, the target and trained result are close with sampling as expected. We note that due to the spectral nature of training there are small oscillations at the end point of x domain, but this can be eliminated by adding an extra regularization that contains information about derivatives. We conclude that DQGM is a versatile tool for cases where distributions follow smooth functions and are expressible and differentiable.

-
- [1] S. Aaronson, *Quantum Computing Since Democritus* (Cambridge University Press, Cambridge, UK, 2013).
- [2] S. Aaronson, The limits of quantum, *Sci. Am.* **298**, 62 (2008).
- [3] B. M. Terhal and D. P. DiVincenzo, Adaptive quantum computation, constant depth quantum circuits and Arthur-Merlin games, *Quant. Inf. Comp.* **4**, 134 (2004).
- [4] S. Boixo, S. V. Isakov, V. N. Smelyanskiy, R. Babbush, Nan Ding, Zhang Jiang, M. J. Bremner, J. M. Martinis, and H. Neven, Characterizing quantum supremacy in near-term devices, *Nat. Phys.* **14**, 595 (2018).
- [5] F. Arute *et al.*, Quantum supremacy using a programmable superconducting processor, *Nature (London)* **574**, 505 (2019).
- [6] H.-S. Zhong *et al.*, Quantum computational advantage using photons, *Science* **370**, 1460 (2020).
- [7] Y. Wu *et al.*, Strong quantum computational advantage using a superconducting quantum processor, *Phys. Rev. Lett.* **127**, 180501 (2021).
- [8] Q. Zhu *et al.*, Quantum computational advantage via 60-qubit 24-cycle random circuit sampling, *Sci. Bull.* **67**, 240 (2022).
- [9] A. W. Harrow and A. Montanaro, Quantum computational supremacy, *Nature (London)* **549**, 203 (2017).
- [10] X. Mi, P. Roushan *et al.*, Information scrambling in computationally complex quantum circuits, *Science* **374**, 1479 (2021).
- [11] M. Benedetti, E. Lloyd, S. Sack, and M. Fiorentini, Parameterized quantum circuits as machine learning models, *Quantum Sci. Technol.* **4**, 043001 (2019).
- [12] M. Cerezo, A. Arrasmith, R. Babbush, S. C. Benjamin, S. Endo, K. Fujii, J. R. McClean, K. Mitarai, Xiao Yuan, L. Cincio, and P. J. Coles, Variational quantum algorithms, *Nat. Rev. Phys.* **3**, 625 (2021).
- [13] M. Born, Zur quantenmechanik der stoßvorgänge, *Z. Phys.* **37**, 863 (1926).
- [14] Z.-Y. Han, J. Wang, H. Fan, L. Wang, and P. Zhang, Unsupervised generative modeling using matrix product states, *Phys. Rev. X* **8**, 031012 (2018).
- [15] M. Benedetti, D. Garcia-Pintos, O. Perdomo, V. Leyton-Ortega, Y. Nam, A. Perdomo-Ortiz, A generative modeling approach for benchmarking and training shallow quantum circuits, *npj Quantum Inf.* **5**, 45 (2019).
- [16] S. Cheng, J. Chen, L. Wang, Information perspective to probabilistic modeling: Boltzmann machines versus born machines, *Entropy* **20**, 583 (2018).

- [17] J.-G. Liu and L. Wang, Differentiable learning of quantum circuit Born machines, *Phys. Rev. A* **98**, 062324 (2018).
- [18] B. Coyle, D. Mills, V. Danos, and E. Kashefi, The Born supremacy: Quantum advantage and training of an Ising Born machine, *npj Quantum Inf.* **6**, 60 (2020).
- [19] J. Zeng, Y. Wu, J.-G. Liu, L. Wang, and J. Hu, Learning and inference on generative adversarial quantum circuits, *Phys. Rev. A* **99**, 052306 (2019).
- [20] C. Zoufal, A. Lucchi, and S. Woerner, Quantum generative adversarial networks for learning and loading random distributions, *npj Quantum Inf.* **5**, 103 (2019).
- [21] Y. Du, M.-H. Hsieh, D. Tao, Efficient online quantum generative adversarial learning algorithms with applications, *Phys. Rev. Lett.* **128**, 110501 (2022).
- [22] S. Lloyd and C. Weedbrook, Quantum generative adversarial learning, *Phys. Rev. Lett.* **121**, 040502 (2018).
- [23] P.-L. Dallaire-Demers and N. Killoran, Quantum generative adversarial networks, *Phys. Rev. A* **98**, 012324 (2018).
- [24] B. Coyle, M. Henderson, J. C. Jin Le, N. Kumar, M. Pains, and E. Kashefi, Quantum versus classical generative modeling in finance, *Quantum Sci. Technol.* **6**, 024013 (2021).
- [25] H.-L. Huang *et al.*, Experimental quantum generative adversarial networks for image generation, *Phys. Rev. Appl.* **16**, 024051 (2021).
- [26] M. Benedetti, E. Grant, L. Wossnig, and S. Severini, Adversarial quantum circuit learning for pure state approximation, *New J. Phys.* **21**, 043023 (2019).
- [27] S. Y. Chang, S. Herbert, S. Vallecorsa, E. F. Combarro, and R. Duncan, Dual-parameterized quantum circuit GAN model in high energy physics, *EPJ Web Conf.* **251**, 03050 (2021).
- [28] B. Øksendal, *Stochastic Differential Equations: An Introduction with Applications* (Springer-Verlag, Berlin/Heidelberg, 1998).
- [29] A. E. Paine, V. E. Elfving, and O. Kyriienko, Quantum quantile mechanics: Solving stochastic differential equations for generating time-series, *Adv Quantum Technol.* **6**, 2300065 (2023).
- [30] O. Kyriienko, A. E. Paine, and V. E. Elfving, Solving nonlinear differential equations with differentiable quantum circuits, *Phys. Rev. A* **103**, 052416 (2021).
- [31] M. Knudsen and C. B. Mendl, Solving differential equations via continuous-variable quantum computers, [arXiv:2012.12220](https://arxiv.org/abs/2012.12220).
- [32] L. Yang, D. Zhang, and G. E. Karniadakis, Physics-informed generative adversarial networks for stochastic differential equations, [arXiv:1811.02033](https://arxiv.org/abs/1811.02033).
- [33] D. Zhang, L. Guo and G. E. Karniadakis, Learning in modal space: Solving time-dependent stochastic PDEs using physics-informed neural networks, [arXiv:1905.01205](https://arxiv.org/abs/1905.01205).
- [34] M. Raissi, Deep hidden physics models: Deep learning of nonlinear partial differential equations, [arXiv:1801.06637](https://arxiv.org/abs/1801.06637).
- [35] J. Gonzalez-Conde, A. Rodriguez-Rozas, E. Solano, and M. Sanz, Efficient Hamiltonian simulation for solving option Price dynamics, *Phys. Rev. Res.* **5**, 043220 (2023).
- [36] S. K. Radha, Quantum option pricing using Wick rotated imaginary time evolution, [arXiv:2101.04280](https://arxiv.org/abs/2101.04280).
- [37] K. Kubo, Y. O. Nakagawa, S. Endo, and S. Nagayama, Variational quantum simulations of stochastic differential equations, *Phys. Rev. A* **103**, 052425 (2021).
- [38] H. Alghassi, A. Deshmukh, N. Ibrahim, N. Robles, S. Woerner, and C. Zoufal, A variational quantum algorithm for the Feynman-Kac formula, *Quantum* **6**, 730 (2022).
- [39] P. Reberstrost, B. Gupt, and T. R. Bromley, Quantum computational finance: Monte Carlo pricing of financial derivatives, *Phys. Rev. A* **98**, 022321 (2018).
- [40] N. Stamatopoulos, D. J. Egger, Yue Sun, C. Zoufal, R. Iten, Ning Shen, and S. Woerner, Option pricing using quantum computers, *Quantum* **4**, 291 (2020).
- [41] A. Martin, B. Candelas, Á. Rodríguez-Rozas, J. D. Martín-Guerrero, Xi Chen, L. Lamata, R. Orus, E. Solano, and M. Sanz, Toward pricing financial derivatives with an IBM quantum computer, *Phys. Rev. Res.* **3**, 013167 (2021).
- [42] S. Chakrabarti, R. Krishnakumar, G. Mazzola, N. Stamatopoulos, S. Woerner, and W. J. Zeng, A threshold for quantum advantage in derivative pricing, *Quantum* **5**, 463 (2021).
- [43] D. J. Egger, C. Gambella, J. Marecek, S. McFaddin, M. Mevissen, R. Raymond, A. Simonetto, S. Woerner, and E. Yndurain, Quantum computing for finance: State-of-the-art and future prospects, *IEEE Trans. Quantum Eng.* **1**, 3101724 (2020).
- [44] P. C. S. Costa, S. Jordan, and A. Ostrander, Quantum algorithm for simulating the wave equation, *Phys. Rev. A* **99**, 012323 (2019).
- [45] A. M. Childs, J.-P. Liu, and A. Ostrander, High-precision quantum algorithms for partial differential equations, *Quantum* **5**, 574 (2021).
- [46] N. Linden, A. Montanaro, and C. Shao, Quantum vs. classical algorithms for solving the heat equation, *Commun. Math. Phys.* **395**, 601 (2022).
- [47] A. M. Childs and J.-P. Liu, Quantum spectral methods for differential equations, *Commun. Math. Phys.* **375**, 1427 (2020).
- [48] M. Lubasch, J. Joo, P. Moinier, M. Kiffner, and D. Jaksch, Variational quantum algorithms for nonlinear problems, *Phys. Rev. A* **101**, 010301(R) (2020).
- [49] S. Wang, Z. Wang, W. Li, L. Fan, G. Cui, Z. Wei, and Y. Gu, A quantum Poisson solver implementable on NISQ devices, [arXiv:2005.00256](https://arxiv.org/abs/2005.00256).
- [50] J. J. García-Ripoll, Quantum-inspired algorithms for multivariate analysis: From interpolation to partial differential equations, *Quantum* **5**, 431 (2021).
- [51] P. García-Molina, J. Rodríguez-Mediavilla, and J. J. García-Ripoll, Quantum Fourier analysis for multivariate functions and applications to a class of Schrödinger-type partial differential equations, *Phys. Rev. A* **105**, 012433 (2022).
- [52] C. B. D. Goes, T. O. Maciel, G. G. Pollachini, R. Cuenca, J. P. L. C. Salazar, and E. I. Duzzioni, QBoost for regression problems: Solving partial differential equations, *Quantum Inf. Process.* **22**, 129 (2023).
- [53] Z. Patel and M. Rummel, Extremal learning: Extremizing the output of a neural network in regression problems, [arXiv:2102.03626](https://arxiv.org/abs/2102.03626).
- [54] J. Romero and A. Aspuru-Guzik, Variational quantum generators: Generative adversarial quantum machine learning for continuous distributions, *Adv. Quantum Technol.* **4**, 2000003 (2021).
- [55] A patent application for the method described in this manuscript has been submitted by Pasqal.
- [56] M. Y. Niu, A. Zlokapa, M. Broughton, S. Boixo, M. Mohseni, V. Smelyanskiy, H. Neven, Entangling quantum generative adversarial networks, *Phys. Rev. Lett.* **128**, 220505 (2022).

- [57] E. Y. Zhu, S. Johri, D. Bacon, M. Esencan, Jungsang Kim, M. Muir, N. Murgai, J. Nguyen, N. Pienti, A. Schouela, K. Sosnova, and K. Wright, Generative quantum learning of joint probability distribution functions, *Phys. Rev. Res.* **4**, 043092 (2022).
- [58] C. Leadbeater, L. Sharrock, B. Coyle, and M. Benedetti, F-Divergences and cost function locality in generative modeling with quantum circuits, *Entropy* **23**, 1281 (2021).
- [59] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, Cambridge, UK, 2010).
- [60] M. Cerezo, A. Sone, T. Volkoff, L. Cincio, and P. J. Coles, Cost function dependent barren plateaus in shallow parametrized quantum circuits, *Nat. Commun.* **12**, 1791 (2021).
- [61] J. Preskill, *Lecture Notes for Ph219/CS219: Quantum Information and Computation*, Chap. 5. https://ilorenz.org/quantumcomputers/literature/preskill_5_update.pdf.
- [62] A. G. Rattew and B. Koczor, Preparing arbitrary continuous functions in quantum registers with logarithmic complexity, [arXiv:2205.00519](https://arxiv.org/abs/2205.00519).
- [63] V. I. Bogachev, N. V. Krylov, M. Röckner, and S. V. Shaposhnikov, *Fokker–Planck–Kolmogorov equations, mathematical surveys and monographs* (American Mathematical Society, Providence, RI, 2015), Vol. 207.
- [64] N. Heim, A. Ghosh, O. Kyriienko, and V. E. Elfving, Quantum model-discovery, [arXiv:2111.06376](https://arxiv.org/abs/2111.06376).
- [65] Y. Du and T. A. Zaki, Evolutional deep neural network, *Phys. Rev. E* **104**, 045303 (2021).
- [66] K. Mitarai, M. Negoro, M. Kitagawa, and K. Fujii, Quantum circuit learning, *Phys. Rev. A* **98**, 032309 (2018).
- [67] M. Schuld, V. Bergholm, C. Gogolin, J. Izaac, and N. Killoran, Evaluating analytic gradients on quantum hardware, *Phys. Rev. A* **99**, 032331 (2019).
- [68] O. Kyriienko and V. E. Elfving, Generalized quantum circuit differentiation rules, *Phys. Rev. A* **104**, 052417 (2021).
- [69] M. Schuld and N. Killoran, Quantum machine learning in feature Hilbert spaces, *Phys. Rev. Lett.* **122**, 040504 (2019).
- [70] T. Goto, Q. H. Tran, and K. Nakajima, Universal approximation property of quantum feature map, *Phys. Rev. Lett.* **127**, 090506 (2021).
- [71] M. Schuld, R. Sweke, and J. J. Meyer, Effect of data encoding on the expressive power of variational quantum-machine-learning models, *Phys. Rev. A* **103**, 032430 (2021).
- [72] M. C. Caro, E. Gil-Fuster, J. J. Meyer, J. Eisert, and R. Sweke, Encoding-dependent generalization bounds for parametrized quantum circuits, *Quantum* **5**, 582 (2021).
- [73] A. Skolik, J. R. McClean, M. Mohseni, P. van der Smagt, and M. Leib, Layerwise learning for quantum neural networks, *Quantum Mach. Intell.* **3**, 5 (2021).
- [74] M. Mottonen, J. J. Vartiainen, V. Bergholm, and M. M. Salomaa, Transformation of quantum states using uniformly controlled rotations, *Quantum Inf. Comput.* **5**, 467 (2005).
- [75] D. Gottesman and I. L. Chuang, Demonstrating the viability of universal quantum computation using teleportation and single-qubit operations, *Nature (London)* **402**, 390 (1999).
- [76] S. Dolgov, K. Anaya-Izquierdo, C. Fox, and R. Scheichl, Approximation and sampling of multivariate probability distributions in the tensor train decomposition, *Stat Comput.* **30**, 603 (2020).
- [77] G. Agliardi and E. Prati, Optimal tuning of quantum generative adversarial networks for multivariate distribution loading, *Quantum Rep.* **4**, 75 (2022).
- [78] M. Hibat-Allah, M. Mauri, J. Carrasquilla, and A. Perdomo-Ortiz, A framework for demonstrating practical quantum advantage: Comparing quantum against classical generative models, *Commun. Phys.* **7**, 68 (2024).
- [79] R. B. Nelsen, *An Introduction to Copulas*, Springer Series in Statistics (Springer, New York, NY, 2006).
- [80] H.-Y. Huang, M. Broughton, J. Cotler, S. Chen, Jerry Li, M. Mohseni, H. Neven, R. Babbush, R. Kueng, J. Preskill, and J. R. McClean, Quantum advantage in learning from experiments, *Science* **376**, 1182 (2022).
- [81] C. Meyer, The Bivariate Normal Copula, *Commun. Stat.-Theory Methods* **42**, 2402 (2013).
- [82] H. J. Choe, C. Ahn, B. J. Kim, Y.-K. Ma, Copulas from the Fokker–Planck equation, *J. Math. Anal. Appl.* **406**, 519 (2013).
- [83] A. Kandala, A. Mezzacapo, K. Temme, M. Takita, M. Brink, J. M. Chow, and J. M. Gambetta, Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets, *Nature (London)* **549**, 242 (2017).
- [84] X.-Z. Luo, J.-G. Liu, P. Zhang, L. Wang, Yao.jl: Extensible, Efficient framework for quantum algorithm design, *Quantum* **4**, 341 (2020).
- [85] P. E. Kloeden and E. Platen, *Numerical Solution of Stochastic Differential Equations* (Springer, Berlin, 1992).
- [86] G. Steinbrecher and W. T. Shaw, Quantile mechanics, *Eur. J. Appl. Math.* **19**, 87 (2008).
- [87] Z. Holmes, K. Sharma, M. Cerezo, P. J. Coles, Connecting ansatz expressibility to gradient magnitudes and barren plateaus, *PRX Quantum* **3**, 010313 (2022).
- [88] L. Bittel and M. Kliesch, Training variational quantum algorithms is NP-hard, *Phys. Rev. Lett.* **127**, 120502 (2021).
- [89] M. S. Rudolph, S. Lerch, S. Thanasilp, O. Kiss, S. Vallecorsa, M. Grossi, and Z. Holmes, Trainability barriers and opportunities in quantum generative modeling, [arXiv:2305.02881](https://arxiv.org/abs/2305.02881).
- [90] M. Larocca, N. Ju, D. García-Martín, P. J. Coles, M. Cerezo, Theory of overparametrization in quantum neural networks, *Nat. Comput. Sci.* **3**, 542 (2023).
- [91] L. Leone, S. F. E. Oliviero, L. Cincio, M. Cerezo, On the practical usefulness of the hardware efficient ansatz, *Quantum* **8**, 1395 (2024).
- [92] A. E. Paine, V. E. Elfving, and O. Kyriienko, Physics-Informed Quantum Machine Learning: Solving nonlinear differential equations in latent spaces without costly grid evaluations, [arXiv:2308.01827](https://arxiv.org/abs/2308.01827).
- [93] C. A. Williams, A. E. Paine, H.-Y. Wu, V. E. Elfving, O. Kyriienko, Quantum chebyshev transform: Mapping, embedding, learning and sampling distributions, [arXiv:2306.17026](https://arxiv.org/abs/2306.17026).
- [94] H.-Y. Wu, V. E. Elfving, O. Kyriienko, Multidimensional quantum generative modeling by quantum hartley transform, [arXiv:2406.03856](https://arxiv.org/abs/2406.03856).