Unsupervised quantum circuit learning in high energy physics

Andrea Delgado[1,*] and Kathleen E. Hamilton[2,†]

[1]*Physics Division, Oak Ridge National Laboratory, Oak Ridge, Tennessee 37830, USA*
[2]*Computer Science and Engineering Division, Oak Ridge National Laboratory, Oak Ridge, Tennessee 37830, USA*

(Received 4 April 2022; accepted 22 September 2022; published 9 November 2022)

Unsupervised training of generative models is a machine learning task that has many applications in scientific computing. In this work we evaluate the efficacy of using quantum circuit-based generative models to generate synthetic data of high energy physics processes. We use nonadversarial, gradient-based training of quantum circuit Born machines to generate joint distributions over two and three variables.

DOI: 10.1103/PhysRevD.106.096006

I. INTRODUCTION

High-energy physics seeks to understand matter at the most fundamental level. Vast and complex accelerators have been built to elucidate the dynamical basis of the fundamental constituents. At these large-scale facilities, high-performance data storage and processing systems are needed to store, access, retrieve, distribute, and process experimental data. Experiments like the Compact Muon Solenoid (CMS) and the A Toroidal LHC Apparatus (ATLAS) at the Large Hadron Collider (LHC) are incredibly complex, involving thousands of detector elements that produce raw experimental data rates over a Tb/sec, resulting in the annual productions of datasets in the scale of hundreds of terabytes to petabytes. In addition, the manipulation of these complex datasets into summaries suitable for the extraction of physics parameters and model comparison is a time-consuming and challenging task.

A crucial element of any analysis workflow in particle physics involves simulating the physical processes and interactions at these facilities to develop new theories and models, explain experimental data, and characterize background. These simulations also allow for studying detector response and plan detector upgrades. The simulation of particle interactions in the detector volume is often computationally intensive, taking up a significant fraction of the computational resources available to physicists.

Recently, alternative methods for detector simulation and data analysis tasks have been explored, like machine learning (ML) applications and quantum information science (QIS). Although ML applications have already been incorporated in high-energy physics (HEP) experimental and data analysis environments, QIS is still an evolving field and its applications and their suitability remain to be explored. QIS is a rapidly developing field focused on understanding the analysis, processing, and transmission of information using quantum mechanical principles and computational techniques. QIS may be able to address the conventional computing gap associated with HEP-related problems, specifically those computational tasks that challenge CPUs and Graphical processor units, such as efficient and accurate event generators. Thus, the relevance of having an efficient simulation mechanism that can faithfully reproduce particle interactions after a high-energy collision has sparked the development of alternative methods. One particular example is the use of generative models such as generative adversarial networks (GANs) [1], which have been utilized in HEP as a tool for fast Monte Carlo (MC) simulations [2–4], and as a machine learning-enhanced method for event generation [5–8]. Some of these studies report up to 5 orders of magnitude decrease in computing time. A crucial feature of generative models is their ability to generate synthetic data by learning from actual samples without knowing the underlying physical laws of the original system. In some studies, generative models have been shown to overcome the statistical limitations of an input sample in subtraction of negative-weight events in samples generated beyond leading order in QCD [5], and to increase the statistics of centrally produced Monte Carlo datasets [7]. Quantum-assisted models have also been proposed for Monte Carlo event generation [9], detector simulation [10,11], and determining the parton distribution functions in a proton [12]. In this work, we successfully trained a quantum generative model to reproduce kinematic distributions of particles in pp interactions at the LHC, with high fidelity. Quantum circuit Born machines (QCBM) trained via gradient-based optimization [13–15] are examples of circuit-based parametrized models that can be trained on near-term quantum platforms.

*delgadoa@ornl.gov
†hamiltonke@ornl.gov

II. MODEL AND LEARNING ALGORITHM

Although generative models trained in adversarial settings have been proven to be a valuable tool in HEP, for this work, we focus on generative models trained with non-adversarial methods. Expressly, we set up and train multiple QCBM using data-driven circuit learning (DDCL) [13–15]. DDCL employs a classical-quantum hybrid feedback loop, as described in Fig. 1.

Generating synthetic data for this HEP application consists of three stages: First, we encode our data of M observations with N-dimensional, numerical features ($\mathcal{D} = \{X^{(1)}, \ldots, X^{(m)}, \ldots, X^{(M)}\}$) into a distribution $P(x)$ defined over finite length bitstrings (x); second, we train a parametrized circuit ansatz using DDCL to prepare an approximation of this distribution $\tilde{P}(x)$; third, we generate synthetic data by decoding bitstrings sampled from $\tilde{P}(x)$. In the following subsection, we describe these stages in greater detail.

A. Data encoding

The kinematic distributions of a particle jet in a high-energy collider experiment such as the LHC can be constructed as a two-dimensional joint distribution [over $(p_T, mass)$ features] or three-dimensional joint distributions [over $(p_T, mass, \eta)$ features]. Each of these kinematic variables is encoded as a discretized binary string using $q = Q/N$ qubits. The marginal distribution is discretized into 2^q bins and each bin index is converted to a 2^q-length binary bitstring.

For a QCBM constructed with Q total qubits, $P(x)$ is constructed by concatenating lists of binary bitstrings which encode the marginals of individual features in a classical dataset. The N-dimensional correlated data features ($X^{(m)}$) are encoded as 2^Q-length binary strings (x_i). Thus, the total Q qubits contain a concatenation of q-dimensional bitstrings. The final 2^Q-length bitstrings are constructed by concatenating the N feature 2^q-length binary bitstrings and normalizing the amplitudes.

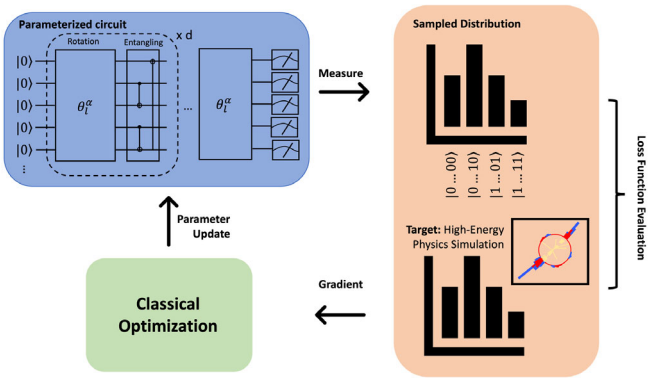

FIG. 1. Diagram of the differentiable QCBM training scheme.

B. Quantum circuit model and training

The QCBM is an example of an implicit model for generative learning [16] that generates data by measuring the system as a Born machine. A QCBM is a parametrized unitary $\mathcal{U}(\Theta)$ that prepares Q qubits in the state $|\psi_\Theta\rangle = \mathcal{U}(\Theta)|\psi_0\rangle$. The initial state $|\psi_0\rangle$ is fixed, and in this study, we use the following: the all-zero state $|\psi_0\rangle = |0\rangle^{\otimes Q}$; a product state of $Q/2$ Bell states $|\Phi^+\rangle^{\otimes Q/2}$; and a product state of $Q/3$ GHZ states $|\psi_0\rangle = |\mathrm{GHZ}\rangle^{\otimes Q/3}$.

Measuring $|\psi_\Theta\rangle$ in a fixed basis $\mathcal{M}$[1] requires sampling from the state with N_{shots} shots. This defines a classical distribution over the 2^Q computational basis states $\tilde{P}_\Theta(x)$ that is used in training, and later used to generate the synthetic data $\tilde{X}$.

1. Parametrized quantum circuit

Finding the ideal unitary $\mathcal{U}(\Theta)$ is dependent on the parametrized quantum circuit (PQC). The PQC design plays an essential role in the performance of many variational hybrid quantum-classical algorithms [17,18] by defining the hypothesis class. For our application, PQCs must be able to model different types of correlations in the input data. This requires circuits to prepare strongly entangled quantum states and the ability to explore Hilbert space.

Variational algorithms have been implemented using quantum circuits composed of a network of single and two-qubit operations, with rotation angles serving as variational parameters. The pattern defining the network of gates is referred to as a *unit cell* or *circuit block* that can be repeated to suit the needs of the application or task at hand. Recently, the term multilayer quantum circuit (MPQC) was coined to describe this type of variational circuit architecture [19]. In this study, we train two circuit templates or *Ansätze*. Each circuit is defined using a Q-qubit register and specified by the number of layers d, consisting of a rotation and an entangling component. *Ansatz* 1 has a combined rotation and entangling gates in a "brick layer" or "simplified two-design" architecture and has been shown to exhibit important properties to study barren plateaus in quantum optimization landscapes [20,21]. *Ansatz* 2 was chosen due to the low correlation displayed between variables and is an extension of *Ansätze* employed in benchmarking tasks [14].

The diagram for one layer of each template is shown in Fig. 2. The rotation gate layers are parametrized by the arbitrary single-qubit rotation gate implemented in PennyLane [22], which has three rotation angles $R(\Theta)^{(i)}_\ell = R(\omega, \theta, \phi)^{(i)}_\ell$, where the layer index ℓ runs from 0 to d, and i is the qubit index.

[1]We use the Z-basis $\mathcal{M} = Z_i \otimes \cdots \otimes Z_Q$ unless otherwise noted.

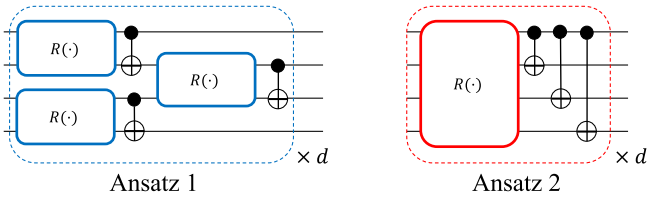

FIG. 2. Diagram for one layer of the circuit templates used to construct and train the QCBM. Both *Ansätze* are constructed using a layered pattern of rotation and entangling gates. For both *Ansätze* a final layer of rotation gates is added before measurement.

We choose to train two *Ansätze* that can be embedded onto near-term noisy intermediate-scale quantum devices: either using a 1D chain of qubits (for *Ansatz* 1), or the "disconnected tree" configuration (for *Ansatz* 2). For *Ansatz* 1 used in this study, the number of parameters to optimize during training is $N_{\text{par}}(N_\ell, Q) = 3N_\ell[2Q - 1(2)] + 3Q$ for QCBM with odd (even) qubits Q. For *Ansatz* 2, $N_{\text{par}}(N_\ell, Q) = 3(N_\ell + 1)Q$, regardless of whether Q is even or odd.

2. Training and cost function

Training a QCBM via DDCL optimizes Θ by minimizing a loss function $\mathcal{L}(P, \tilde{P}_\Theta)$ such that $\tilde{P}_\Theta(x) \approx P(x)$. We use the Jensen-Shannon (JS) divergence—a differentiable function that compares two distributions $P(x)$ and $\tilde{P}_\Theta(x)$:

$$JS(P|\tilde{P}_\Theta) = \frac{1}{2}\sum_x \left[P \log\left(\frac{P}{M}\right) + \tilde{P}_\Theta \log\left(\frac{\tilde{P}_\Theta}{M}\right)\right], \quad (1)$$

where $M = (P + \tilde{P}_\Theta)/2$. The minimum value of the loss function $JS(P|\tilde{P}_\Theta) = 0$ is achieved when $\tilde{P}_\Theta(x) = P(x)$. The model parameters Θ are optimized using classical gradient descent methods so that the loss function is minimized. The gradient of the loss function is computed with respect to the circuit parameters using the parameter shift rule [23]. Each QCBM is trained using the ADAM gradient-based optimizer [24] available in the PennyLane library [22].

C. Measurement decoding and postprocessing

The output of a QCBM is $\tilde{P}_\Theta(x)$: a classical distribution over 2^Q-length binary strings. To convert $\tilde{P}_\Theta(x)$ to numerical n-dimensional features ($\tilde{\mathcal{D}} = \{\tilde{X}^{(1)}, \ldots, \tilde{X}^{(m)}, \ldots\}$), we reverse the steps described in Sec. II A: each binary 2^Q-length string is disassociated into N composite strings each of length 2^q, and a float value randomly drawn from a uniform distribution defined with the bin edges previously used to map the samples x_m into the binary basis to generate $P(x)$.

III. VALIDATION ON LHC DATASET

One of the big computational challenges in HEP is the considerable computing time required to model the behavior of subatomic particles both at the vertex and detector level. In Sec. II, we introduce the architecture of a quantum generative model that aims to provide an alternative to traditional Monte Carlo (MC) methods in the context of data augmentation. Thus, to validate the proposed model, we consider the simulation of the production of pairs of jets in pp interactions at the LHC. The dataset [6] consists of 10×10^6 dijet events generated using MADGRAPH 5 version 2.6.4 [25] and PYTHIA8 version 8.307 [26], corresponding to a center of mass energy of 13 TeV and an integrated luminosity of about 0.5 fb^{-1}. The response of the detector was simulated by a DELPHES version 3.4.3pre06 [27] fast simulation, using settings that resemble the ATLAS detector. An average of 25 additional soft-QCD pp collisions (pileup) were added to the simulation to mimic the conditions of a typical collider event realistically. Jets were reconstructed using the anti-k_T [28] algorithm as implemented in FASTJET [29], with a distance parameter $R = 1.0$. A selection cuts on the scalar sum of the transverse momenta of the outgoing partons HT ≥ 500 GeV was applied, reducing the sample size to about 4×10^6 events. The kinematic distributions of the leading jet on the dijet system are used to validate the QCBM models and evaluate its performance in a real-world application.

IV. RESULTS

In this section, we report on the results of training a QCBM to prepare the target distribution of the dataset described in Sec. III. The model performance is evaluated by studying the JS divergence value throughout the training and comparing target (MC expectation) and generated (the output of the trained QCBM model) distributions. We explore the encoding of the target distributions for 2(3) variable joint distributions into 8(12) qubit systems. This scheme allows for a four-qubit encoding per distribution. Unless noted, each marginal distribution is encoded in a target state binned over $2^4 = 16$ basis states. We trained the QCBM circuits in the absence of noise to obtain a set of optimal parameters Θ. Then, the circuits were deployed using the trained parameters on IBM quantum devices to study the effect of noise in the loss landscape, reproducing the target distribution. Finally, a local parameter tuning scheme was applied to improve the performance in the presence of noise.

A. Training with noiseless qubits

1. 2D distributions

In Fig. 3, the JS divergence values are plotted as a function of training step. The circuits were trained using $N_{\text{shots}} = 8192$ to prepare a joint 2D distribution corresponding to the

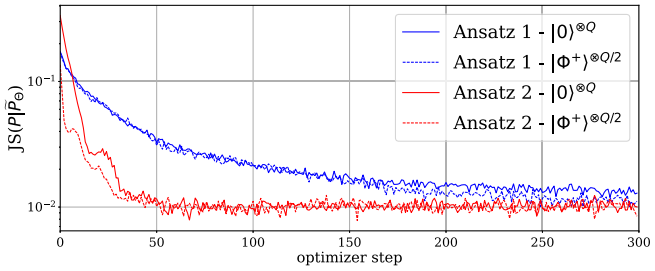

FIG. 3. JS divergence value as a function of training step using $N_{\text{shots}} = 8192$. Blue (red) points correspond to *Ansatz* 1 (2). Circuits were initialized in the all-zero state (solid lines) or four Bell states (dashed lines) and trained to learn a 2D joint distribution.

marginal distributions of the leading jet transverse momentum (p_T) and mass, both binned over the 16 basis states corresponding to four qubits. Circuits were constructed using the *Ansatz* configurations shown in Fig. 2 and trained to start from either the all-zero state ($|\psi_0\rangle = |0\rangle^{\otimes 8}$) or from a product of four Bell states ($|\Phi^+\rangle^{\otimes 4}$). We fixed the number of layers $N_{\text{layers}} = 6$ for *Ansatz* 1 (blue) and 12 for *Ansatz* 2 (red). Each circuit is trained for 300 steps of ADAM with a learning rate $\alpha = 0.01$.

From Fig. 3 we can conclude that *Ansatz* 2 converges to a stable JS divergence value much faster than *Ansatz* 1, and the training of the QCBM is not affected by the choice of the initial state. Figure 4 displays the distributions of samples generated via projective measurements on the qubits in the trained circuits. We observe that the data generated resembles the target distributions with high fidelity, with a slightly better agreement for data generated by sampling from the QCBM constructed with *Ansatz* 2 (red).

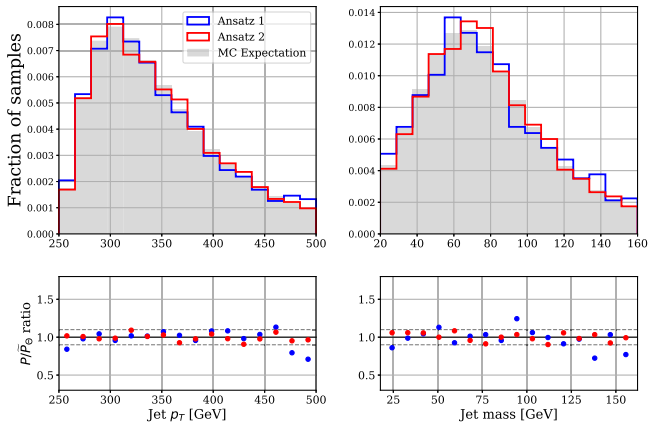

FIG. 4. Top: target and sampled distributions. Blue (red) points correspond to *Ansatz* 1 (2). Circuits were initialized in the all-zero state ($|\psi_0\rangle = |0\rangle^{\otimes 8}$) and trained to learn a 2D joint distribution. Bottom: ratio of target and sampled distributions with horizontal guide lines marking $P/\tilde{P}_\Theta = 0.9$ and $P/\tilde{P}_\Theta = 1.1$.

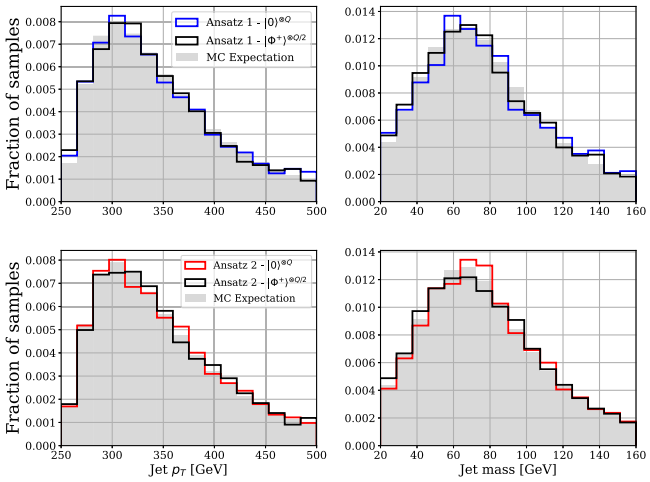

FIG. 5. Top: target and sampled distributions constructed using *Ansatz* 1. Circuits were initialized in the all-zero state (blue) and four Bell states (black). Bottom: target and sampled distributions constructed using *Ansatz* 2. Circuits were initialized in the all-zero state (red) and four Bell states (black).

In Fig. 5, we compare the sampled and target distributions obtained when starting the training from either an all-zero state ($|\psi_0\rangle = |0\rangle^{\otimes 8}$) or from a product of four Bell states ($|\Phi^+\rangle^{\otimes 4}$). We define a similarity measure to perform a systematic comparison of the marginal distributions for each feature by computing the mean absolute error (MAE) per bin between all normalized target and sampled marginal distributions:

$$D(p|\tilde{p}(\Theta)) = \frac{1}{N 2^q} \sum_n^N \sum_i^{2^q} |p_i^{(n)} - \tilde{p}_i^{(n)}(\Theta)|. \quad (2)$$

From both Fig. 5 and Table I, we can conclude that *Ansatz* 2, initialized in the all-zero state, reproduces the feature marginals with the highest fidelity. On the other hand, the difference in $D(p|\tilde{p}(\Theta))$ for the two initial configurations considered is negligible, considering a statistical error proportional to $1/\sqrt{N_{\text{samples}}} \sim 0.0005728$, implying that the training is independent of circuit initialization.

Another important factor in the process of building the circuit to prepare the target state is the number of layers the template or *Ansätze* in Fig. 2 is repeated. This choice will

TABLE I. $D(p|\tilde{p}(\Theta))$ for eight-qubit QCBM.

Ansatz	N_{par}	Initial state	$D(p\|\tilde{p}(\Theta))$
1	276	$\|\psi_0\rangle = \|0\rangle^{\otimes 8}$	0.007153
1		$\|\psi_0\rangle = \|\Phi^+\rangle^{\otimes 4}$	0.006767
2	312	$\|\psi_0\rangle = \|0\rangle^{\otimes 8}$	0.005452
2		$\|\psi_0\rangle = \|\Phi^+\rangle^{\otimes 4}$	0.005696

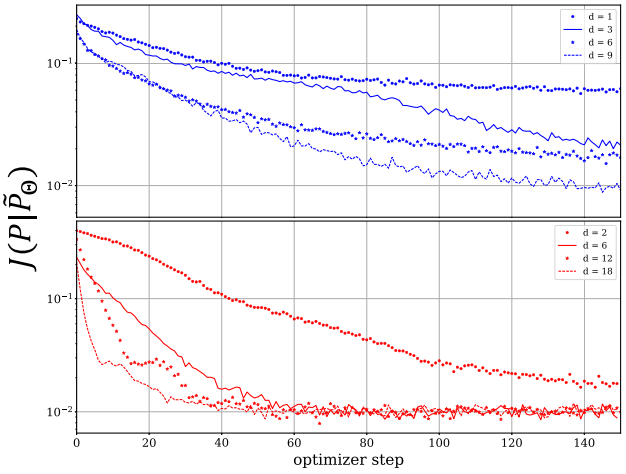

FIG. 6. Loss as a function of training step using $N_{\text{shots}} = 8192$. The different curves represent a different number of layers d, used to construct the eight-qubit QCBM. Top: *Ansatz* 1 (blue). Bottom: *Ansatz* 2 (red).

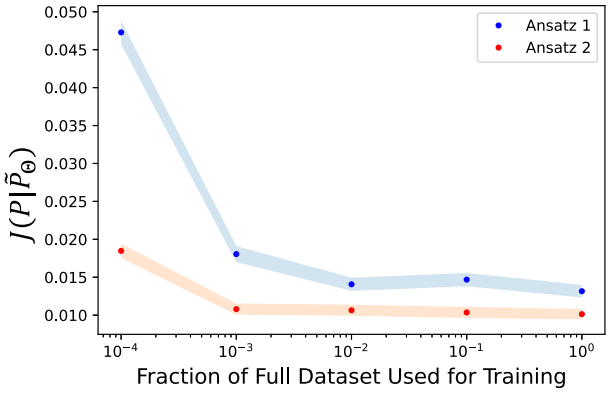

FIG. 7. Mean loss between the complete dataset and the distribution generated by eight-qubit QCBM circuits plotted as solid circles in blue (red) for *Ansatz* 1 (2). Each QCBM model was trained on partial data. Shaded regions correspond to mean JS divergence value ± one standard deviation.

also determine the number of trainable parameters in the model. In Fig. 6, the JS divergence value is plotted as a function of training steps for $d = \{1, 3, 6, 9\}$ layers in *Ansatz* 1; and $d = \{2, 6, 12, 18\}$ layers in *Ansatz* 2. We can see from this plot that the number of layers used to construct *Ansatz* 2 has little impact on the minimal JS value reached after the training converged. Nonetheless, it affects how fast the model reaches this minimal JS value. On the other hand, the training performance of *Ansatz* 1 is highly dependent on the number of layers used to construct the circuit. We chose to use $d = 6(12)$ to construct our QCBM with *Ansatz* 1 (2) to keep an optimal balance between training time and performance.

This study proposes using quantum generative models as a data augmentation tool. The ability of generative models in increasing the statistical precision of the simulated events beyond the training sample has been studied in Refs. [30,31]. The results presented in this section thus far were obtained by training QCBMs on a target distribution using close to 4×10^6 events. We also investigated how reducing the training dataset size affects the trained model's fidelity to reproduce the target distribution. In Fig. 7, the horizontal axis represents the fraction of the initial training dataset of 4×10^6 events used to train the QCBM. Once the model is trained, using a fraction of the full dataset, the JS divergence metric is evaluated on the target distribution generated using the whole dataset. The distribution was obtained by evaluating the QCBM with the trained parameters with 8192 shots. The sampling process was repeated 1000 times, and the mean is reported in Fig. 7 as a solid blue (red) dot for *Ansatz* 1 (2). The bands correspond the mean JS divergence value $\pm\sigma$.

Finally, we report on the correlation matrix between the jet p_T and mass variables used to construct the target distribution. In Table II(a), the values associated with the target distribution are displayed in black. If the trained QCBM learned the joint distribution, one would expect to recover the correlation matrix when evaluating the QCBM with the trained parameters. The results in Table II indicate that this is true for the QCBM constructed with *Ansatz* 1 (red), but not for the QCBM constructed with *Ansatz* 2. The correlation matrix for the latter case indicates that there is little correlation between the marginal distributions in the synthetic samples.

TABLE II. Correlation matrices between jet p_T and mass (m) variables in the (a) target distribution, and samples obtained from the evaluation of the QCBMs constructed using (b) *Ansatz* 1 and (c) *Ansatz* 2 in Fig. 2 with the trained parameters. Values displayed for initial states prepared in the all-zero state (bold) and a product of four Bell states (italics).

(a) Monte Carlo (ground truth)

	p_T	Mass
p_T	⋯	0.2
Mass	0.2	⋯

(b) *Ansatz* 1

	p_T		Mass	
	$\|0\rangle^{\otimes 8}$	$\|\Phi^+\rangle^{\otimes 4}$	$\|0\rangle^{\otimes 8}$	$\|\Phi^+\rangle^{\otimes 4}$
p_T	⋯		**0.19**	0.12
Mass	**0.19**	0.12	⋯	

(c) *Ansatz* 2

	p_T		Mass	
	$\|0\rangle^{\otimes 8}$	$\|\Phi^+\rangle^{\otimes 4}$	$\|0\rangle^{\otimes 8}$	$\|\Phi^+\rangle^{\otimes 4}$
p_T	⋯		-1.0×10^{-3}	-9.1×10^{-3}
Mass	$\mathbf{-1.0 \times 10^{-3}}$	-9.1×10^{-3}	⋯	

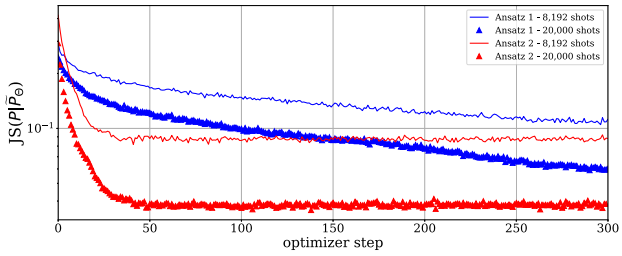

FIG. 8. JS divergence as a function of training step using $N_{\text{shots}} = 8192$ (solid lines) and $N_{\text{shots}} = 20{,}000$ (triangles). Blue (red) points correspond to *Ansatz* 1 (2). Circuits were initialized in the all-zero state and trained to learn a 3D joint distribution. QCBM were initialized in the all-zero state ($|\psi_0\rangle = |0\rangle^{\otimes 12}$.

2. 3D distributions

To understand how the trainability of nonadversarial generative models scales with the number of quantum registers, we increased the number of qubits in our model from 8 to 12. This increment translates into a larger number of basis states ($2^8 = 256$ to $2^{12} = 4096$). Furthermore, the joint probability distribution that we encode in the target state is now three dimensional by including an additional marginal distribution associated with the "forwardness" of the jet with respect to the beam (jet η). In Fig. 8, the JS divergence loss is plotted as a function of training step. The circuits were initialized in the all-zero state ($|\psi_0\rangle = |0\rangle^{\otimes 12}$) and $d = 6(12)$ to construct our QCBM with *Ansatz* 1 (2). In this plot, we can also see the effect of increasing the number of shots during training, reporting a significant difference in JS divergence values when increasing N_{shots} from 8,192 to 20,000.

When training QCBM with $Q = 12$ qubits, we also used an initial state ($|\psi_0\rangle = |\text{GHZ}\rangle^{\otimes Q/3}$) where three-qubit subsets are initialized in a Greenberger–Horne–Zeilinger (GHZ) state. The comparison for the JS divergence value as a function of the training step from the three initial states considered is displayed in Fig. 9. For *Ansatz* 1 (blue), the JS value for the three configurations is very similar for the first 100 training steps. From step 100 on, the QCBM initialized

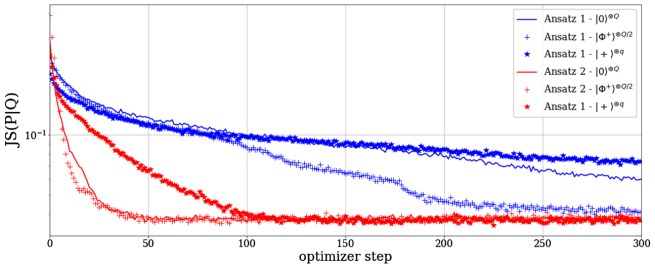

FIG. 9. JS divergence as a function of training step using $N_{\text{shots}} = 20{,}000$. Blue (red) points correspond to *Ansatz* 1 (2). Circuits were initialized in the all-zero state (solid lines), a product of Bell states (cross), or the all-plus state (stars), and trained to learn a 3D joint distribution.

in a product of Bell states (blue cross) trains faster and converges to a lower JS value than the QCBMs initialized in the all-zero (solid blue) and all-plus (blue star) state. For *Ansatz* 2, the effect of the initial state used in the preparation of the circuit has a more negligible effect on the minimal JS value the model converges after training. Nonetheless, the QCBMs prepared from the all-plus state (stars) seem to take longer to converge.

In Fig. 10, we compare the distributions of samples generated via projective measurements on the circuits evaluated on the trained parameters. The circuits were initialized in the all-zero state. By looking at Fig. 10 and Table III, we observe a degraded performance in terms of similarity metric [Eq. (2)] when compared to the eight-qubit circuit results. The $D(p|\tilde{p}(\Theta))$ value increases from 0.007153 to 0.02226 for *Ansatz* 1 and from 0.005452 to 0.0108 for *Ansatz* 2. Again, the trained QCBM that prepares the target distribution with the highest fidelity is *Ansatz* 2.

In Fig. 11, we compare the distributions generated by QCBMs initialized in the three different initial configurations. Again, we see little dependence on the initial state for QCBMs prepared using *Ansatz* 2. Nonetheless, the effect of the initial state in QCBMs prepared using *Ansatz* 1 is now more evident.

Finally, we report on the correlation matrix between the jet p_T, mass and η variables used to construct the target distribution. In Table IV, the values associated with the target distribution are displayed in black. The results in Table IV display a slight discrepancy in the original correlation matrix (training dataset) and that for the samples generated by sampling *Ansatz* 1 (blue) with the trained parameters. Again, for the QCBM constructed with *Ansatz* 2, the matrix values indicate that there is little correlation between the marginal distributions in the synthetic samples.

B. Noisy training with layerwise coordinate descent

The gradient-based training of the QCBM models presented in Sec. IV A was executed on noiseless (ideal) qubits but was not used for training on noisy hardware. However, the performance of the trained QCBM model on near-term quantum devices will be heavily impacted by hardware noise. Qubit initialization, gate noise, and errors in the circuit measurement step all result in state preparation error that can cause parametrized models to converge to maximally mixed states, with an overall effect of flattening the loss landscape [32] and manifests in a noisy estimation of $\tilde{P}_\Theta(x)$.

While developing error mitigation methods that can be incorporated into variational training algorithms is an open area of research, one approach to noise mitigation is to implement the circuit training with hardware noise in order to learn optimized parameters that can compensate for time-independent errors, such as over- and under-rotation in single qubit gates. We test the robustness of the final

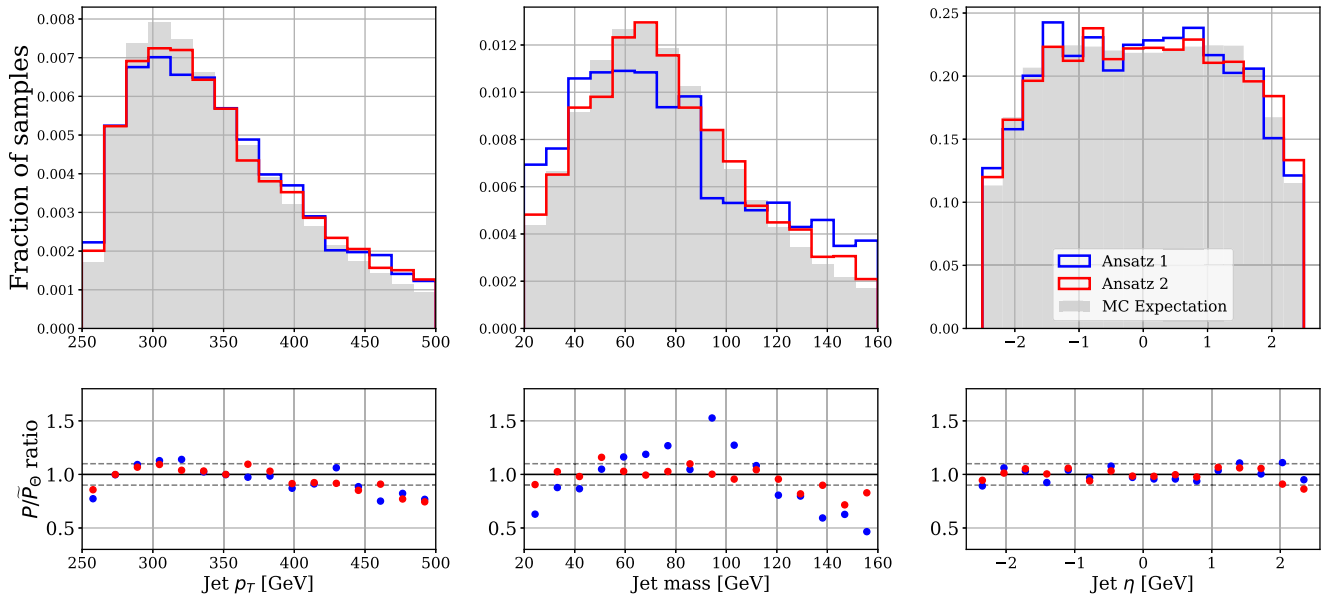

FIG. 10. Top: target and sampled distributions. Blue (red) points correspond to *Ansatz* 1 (2). Circuits were initialized in the all-zero state ($|\psi_0\rangle = |0\rangle^{\otimes 12}$) and trained to learn a 3D joint distribution. Bottom: ratio of target and sampled distributions with horizontal guide lines marking $Q/P = 0.9$ and $Q/P = 1.1$.

parameters found in Sec. IV A to hardware noise by first executing the trained QCBM models constructed with *Ansatz* 2 on superconducting qubit devices. The eight-qubit QCBM (276/312 parameters) was executed on the 16-qubit device `ibmq_guadalupe` and the 12-qubit QCBM (468) was executed on the 27-qubit device `ibm_cairo`; both were accessed through a cloud-based queue.

In the absence of error and noise mitigation we instead opted to use a localized search over individual parameters. Each parametrized *Ansatz* (shown in Fig. 2) is constructed with layers of parametrized rotation gates, implemented using an arbitrary unitary gate with three rotational parameters. Then, we used a layerwise coordinate descent (LCD) method to optimize each QCBM performance on hardware. The workflow is shown in Fig. 12. No readout error mitigation or other noise mitigation methods were used. LCD optimizes the parameters of a circuit $\mathcal{U}(\Theta)$ by searching the multidimensional parameter space along linear cuts of finite width. With N qubits in the register, each parameter was swept through a shift of parameters defined by $\epsilon = -\pi/4$ and spacing $2\epsilon/n$. The targeted backends allowed for a maximum number of circuits per batch (B) which defines the spacing $n = B/N$. For the eight-qubit QCBM trained on `ibmq_guadalupe` this resulted in a grid spacing of 0.0419. For the 12-qubit QCBM trained on `ibm_cairo` this resulted in a mesh spacing of 0.0628. Each circuit was sampled using $N_{\text{shots}} = 20000$. The rotational parameters are optimized starting with the gates closest to the measurement process.

TABLE III. Comparison between target and sampled distributions according to Eq. (2).

Ansatz	N_{par}	Initial state	$D(p\|\tilde{p}(\Theta))$
1	432	$\|\psi_0\rangle = \|0\rangle^{\otimes 12}$	0.0226
1		$\|\Phi^+\rangle^{\otimes 6}$	0.0138
1		$\|\psi_0\rangle = \|\text{GHZ}\rangle^{\otimes 3}$	0.0187
2	468	$\|\psi_0\rangle = \|0\rangle^{\otimes 12}$	0.0108
2		$\|\Phi^+\rangle^{\otimes 6}$	0.0090
2		$\|\psi_0\rangle = \|\text{GHZ}\rangle^{\otimes 3}$	0.0106

The top of Fig. 13 displays the quartiles of JS loss over each iteration of LCD with eight-qubit QCBM circuits. The quartiles are plotted for the QCBM constructed with *Ansätze* 1 and 2 and evaluated with the updated parameters after each iteration on the `ibmq_guadalupe` backend in blue and red, respectively. The plot also shows how the JS divergence value degrades as the LCD training parameters deviate from those obtained during the noiseless training. On the other hand, hardware performance is relatively stable. The bottom of Fig. 13 shows the sampled distributions generated by the evaluation of the QCBM with the parameters that yielded the lowest JS divergence value during the LCD training. We observe a more significant discrepancy between the target and sampled distributions compared to the results reported in Fig. 4, where the QCBM is evaluated with the parameters obtained during the noiseless training. In Fig. 14, the quartiles of JS loss (top) and the sampled and target distributions (bottom) are displayed. The QCBMs are constructed using *Ansatz* 2, to prepare a 3D joint distribution in a 12-qubit register. The LCD training was performed on the `ibm_cairo` device.

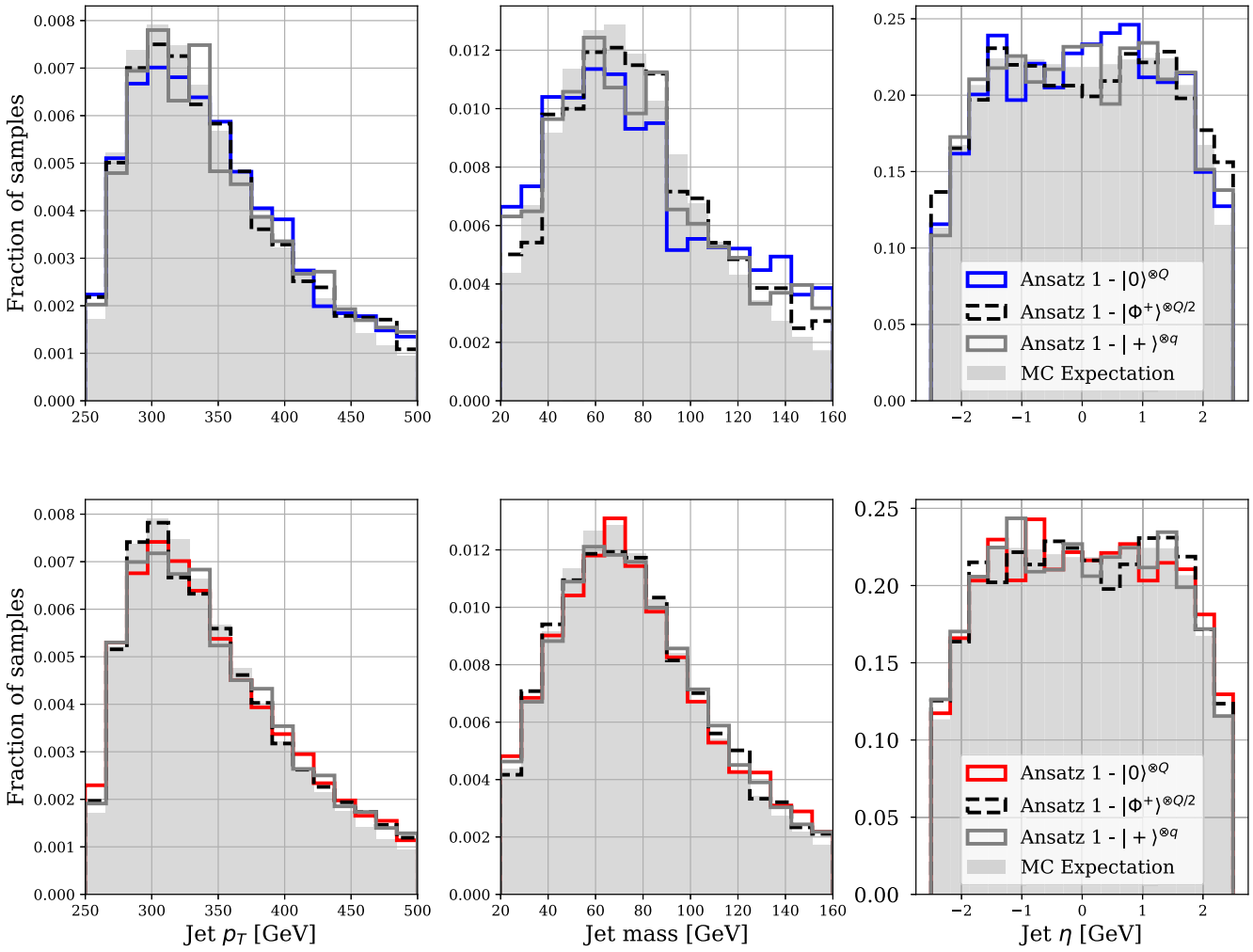

FIG. 11. Top: target and sampled distributions corresponding to *Ansatz* 1. Circuits were initialized in the all-zero state (solid blue line), a product of Bell states (black dashed line), and the all-plus state (gray dashed line) and trained to learn a 3D joint distribution. Bottom: target and sampled distributions corresponding to *Ansatz* 2. Circuits were initialized in the all-zero state (solid red line), a product of Bell states (black dashed line), and the all-plus state (gray dashed line).

In Fig. 15 we show the quartiles of JS for three iterations representative of the different stages of the LCD training (left). During the first iterations in the training, the loss landscape is very flat, and the JS value is lower for the QCBMs evaluated on the `qasm_simulator`. The box plots on the left correspond to the JS values obtained during the parameter sweep in the range $\epsilon \in [-\frac{\pi}{4}, \frac{\pi}{4}]$ for a particular rotation gate in a given layer. Then we observe that, for subsequent iterations, the parameter tuning moves the parameters into regions where the performance of the noiseless simulator is degraded, and slowly improving the performance on the quantum device.

As we can see from Figs. 13 and 14, noisy training can improve performance, but there are a number of obstacles. When the training is initialized with parameters pretrained with noiseless qubits, multiple interactions may be needed to retrain. The gradual improvements in the loss function may not be robust against large deviations in the device noise. For example, executing the LCD workflow over all rotational parameters of the 12-qubit QCBM was done over multiple days, probably causing the discrete change in the loss between iterations 31 and 32.

V. QCBM DESIGN SPACE

Parametrized circuit *Ansätze* used for variational algorithms need to balance expressability with trainability. Characterizing expressability is a difficult problem, and the most common analysis relies on computing frame potentials which compare the distributions of states that a particular ansatz can generate, to the Haar random distribution [17,18]. There are considerable ongoing efforts in determining the characteristics of circuits that lead to effective training and scaling [33]. In this paper we used two different ansatz designs and multiple initializations for the Q-qubit register. In this section we discuss some

TABLE IV. Correlation matrices between jet p_T, mass (m), and η variables in the (a) target distribution, and samples obtained from the evaluation of the QCBMs constructed using (b) *Ansatz* 1 and (c) *Ansatz* 2 in Fig. 2 with the trained parameters. Values displayed for initial states prepared in the all-zero state (bold), a product of six Bell states (italics), and a product of three GHZ states.

(a) Monte Carlo (ground truth)

	p_T	Mass	η
p_T	⋯	0.2	7.3×10^{-12}
Mass	0.2	⋯	2.7×10^{-11}
η	7.3×10^{-12}	2.7×10^{-11}	⋯

(b) *Ansatz* 1

	p_T			Mass (m)			η		
	$\lvert 0\rangle^{\otimes 12}$	$\lvert\Phi^+\rangle^{\otimes 6}$	$\lvert GHZ\rangle^{\otimes 3}$	$\lvert 0\rangle^{\otimes 12}$	$\lvert\Phi^+\rangle^{\otimes 6}$	$\lvert GHZ\rangle^{\otimes 3}$	$\lvert 0\rangle^{\otimes 12}$	$\lvert\Phi^+\rangle^{\otimes 6}$	$\lvert GHZ\rangle^{\otimes 3}$
p_T		⋯		**0.16**	*0.16*	0.18	8.2×10^{-3}	1.2×10^{-3}	-1.3×10^{-3}
m	**0.16**	*0.16*	0.18		⋯		−0.014	4.4×10^{-3}	7.7×10^{-3}
η	8.2×10^{-3}	1.2×10^{-3}	-1.3×10^{-3}	−0.014	4.4×10^{-3}	7.7×10^{-3}		⋯	

(c) *Ansatz* 2

	p_T			Mass (m)			η		
	$\lvert 0\rangle^{\otimes 12}$	$\lvert\Phi^+\rangle^{\otimes 6}$	$\lvert GHZ\rangle^{\otimes 3}$	$\lvert 0\rangle^{\otimes 12}$	$\lvert\Phi^+\rangle^{\otimes 6}$	$\lvert GHZ\rangle^{\otimes 3}$	$\lvert 0\rangle^{\otimes 12}$	$\lvert\Phi^+\rangle^{\otimes 6}$	$\lvert GHZ\rangle^{\otimes 3}$
p_T		⋯		-4.1×10^{-3}	4.6×10^{-3}	0.019	-3.7×10^{-3}	-9.1×10^{-3}	1.6×10^{-3}
m	-4.1×10^{-3}	4.6×10^{-3}	0.019		⋯		6.3×10^{-3}	4.9×10^{-3}	2.2×10^{-3}
η	-3.7×10^{-3}	-9.1×10^{-3}	1.6×10^{-3}	6.3×10^{-3}	4.9×10^{-3}	2.2×10^{-3}		⋯	

observations based on our results reported in Secs. IV A and IV B. The QCBM models we trained in this paper used the same parametrization (the arbitrary rotation gate implemented in PennyLane) and entangling gate operation (the CNOT gate). Each feature was encoded into the same number of qubits (four) which defined the size of the overall register: eight qubits for 2D distributions, 12 qubits for 3D distributions.

Between the two *Ansatz* designs, only *Ansatz* 1 can generate arbitrary Q-qubit entanglement and fit arbitrary

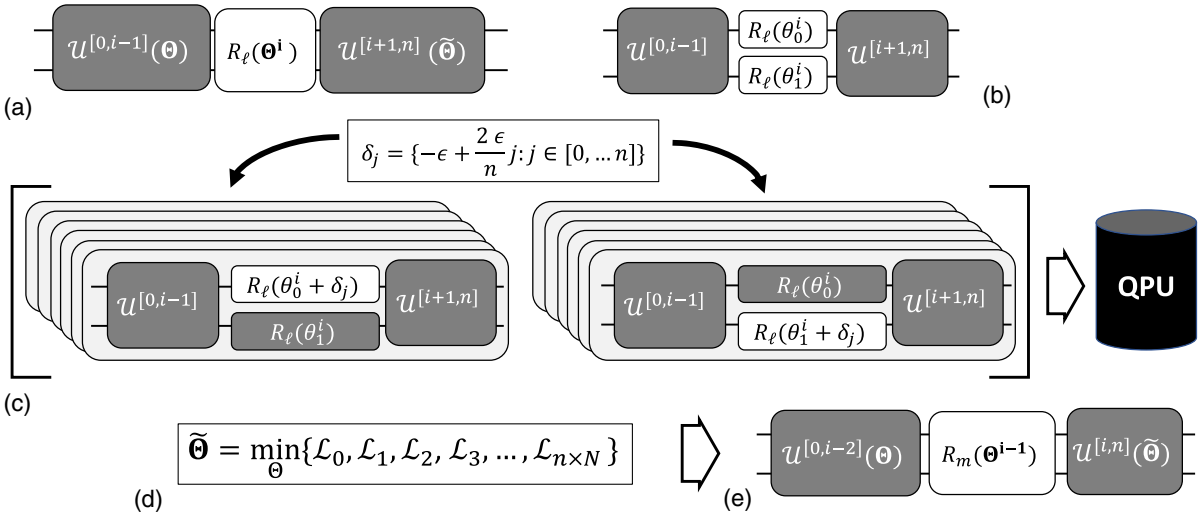

FIG. 12. Layerwise coordinate descent (LCD) workflow: (a) a single rotational layer in a N-qubit QCBM model, (b) is composed of rotational gates R_ℓ acting on individual qubits. A batch of $n \times N$ circuits is constructed by sweeping each individual gate over a discrete set of n shifts and executed on a quantum processor (QPU). The loss is evaluated for each circuit executed (d), the parameter vector Θ is updated by the values which return the minimal value, and the updated parameter vector is used to start the search over the next rotational layer (e).

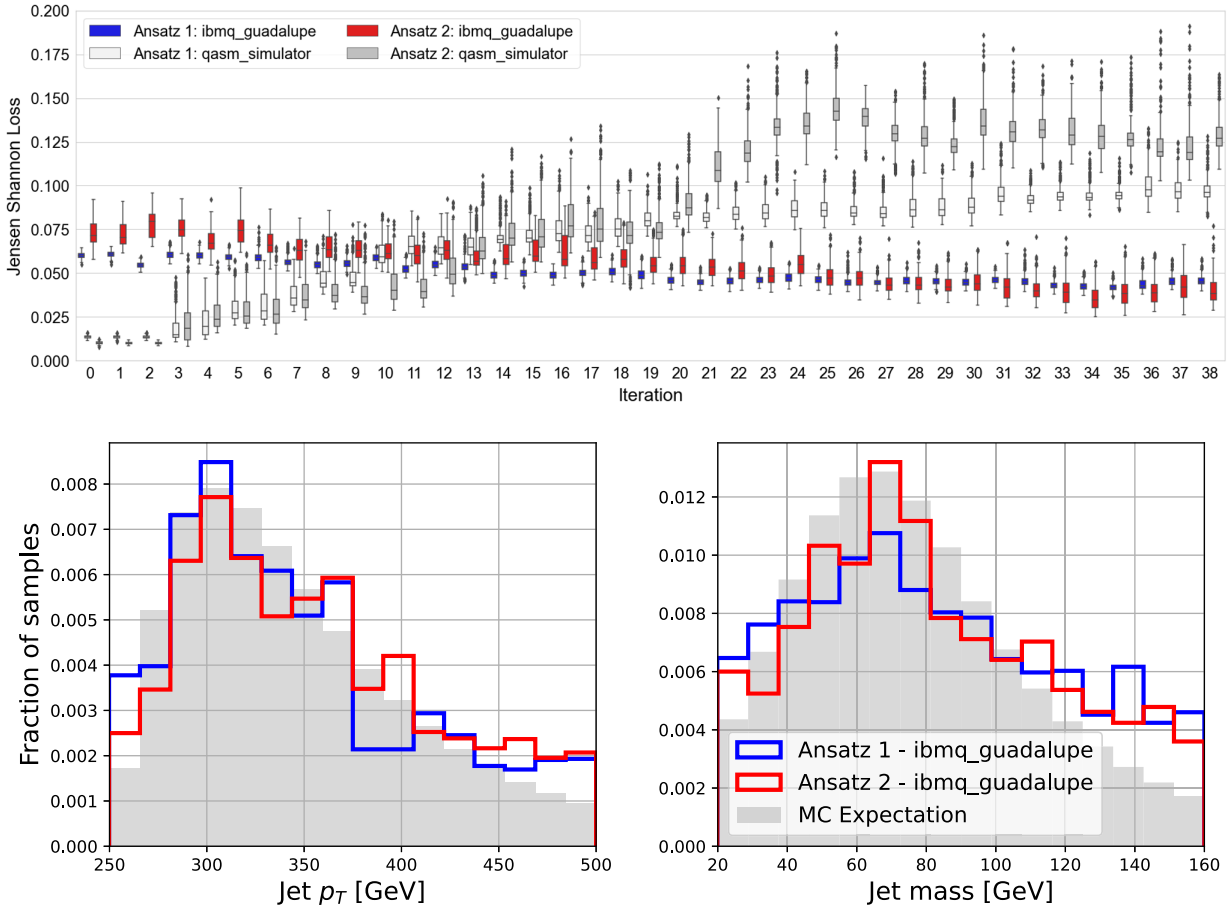

FIG. 13. Top: quartiles of Jensen-Shannon loss over each iteration of LCD with eight-qubit QCBM circuits. Bottom: target and sampled distributions constructed using *Ansatz* 1 (2) in blue (red). Circuits were initialized in the all-zero state and evaluated on `ibmq_guadalupe` with an eight-qubit QCBM. Distributions for the best parameters found after the execution of the LCD workflow.

correlations between two or three variables, regardless of the choice of initial state. However, as shown in Figs. 3, 5, 8, and 11, this ansatz slowly learns. On the other hand, *Ansatz* 2 quickly learns, but only prepares a product state of four-qubit systems. When the qubit register is initialized in the all-zero state $|0\rangle^{\otimes Q}$ or as a set of Bell states $|\Phi^+\rangle^{\otimes Q}$, then *Ansatz* 2 cannot, by definition, model arbitrary correlations between each four-qubit subset. For $Q = 12$, if the circuit is initialized with $|\mathrm{GHZ}\rangle^{\otimes 4}$, then there is local entanglement between the qubit subsets. Yet as reported in Tables II and IV, this is insufficient to capture the correlations as seen in the Monte Carlo data. We observe a trade-off in the modeling capacity of *Ansatz* 1 and *Ansatz* 2: *Ansatz* 1 can model the correlations between variables but has lower fidelity in fitting marginal distributions [as quantified by Eq. (2) in Tables I and III and seen in Figs. 4 and 10]. On the other hand, for *Ansatz* 2 the generated data fails to capture the correlations in the Monte Carlo data, but has high fidelity in fitting marginal distributions (as reported in Tables II and IV). Simply including local correlations in the initial state by using $|\mathrm{GHZ}\rangle^{\otimes 4}$ was insufficient to generate high correlations between variables p_T and mass.

VI. CONCLUSION

The size of the design space associated with parametrized quantum circuit models is large. This work demonstrated the efficacy of nonadversarial unsupervised training of generative models implemented as parametrized quantum circuits. We demonstrate the usefulness of these quantum models in the context of a HEP application and to assist other practitioners, we have used several circuit *Ansätze* found in the quantum computing literature and tested the trainability of QCBM initialized with different quantum states. We quantify the fidelity of the trained models using the JS score (also used to train the models), the MAE of feature marginals, and the correlation matrices of generated data.

We are encouraged by the success of gradient-based training for 12-qubit QCBM. We show that for two and three correlated variables, both *Ansätze* can minimize the loss to the order of $\sim 10^{-2}$, but whether that corresponds to

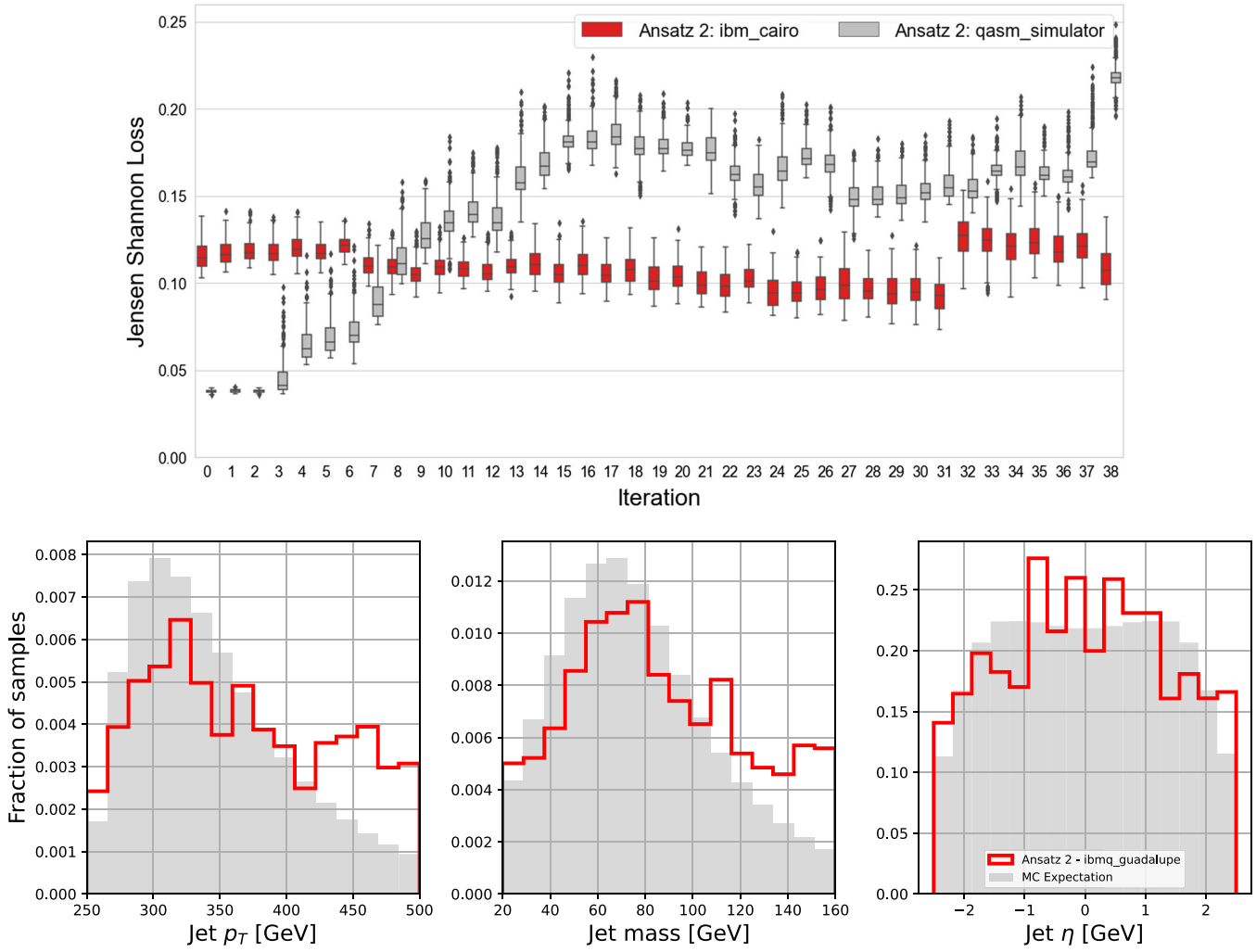

FIG. 14. Top: quartiles of Jensen-Shannon loss over each iteration of LCD implemented in `ibm_cairo` with a 12-qubit QCBM. Bottom: target (gray) and sampled distributions constructed using *Ansatz* 2. Circuits were initialized in the all-zero state and evaluated on `ibm_cairo` with an 12-qubit QCBM. Distributions for the best parameters were found after the execution of the LCD workflow.

models that can faithfully reproduce the kinematic distributions of a jet in a pp collision typical of the LHC experiment cannot be deduced by the training loss alone. Only *Ansatz* 1 can fit the correlations between variables in the absence of noise, but the individual marginal fits for *Ansatz* 1 are lower fidelity than *Ansatz* 2. On the other hand, while *Ansatz* 2 can reproduce the individual marginals with high fidelity it cannot, by definition, model correlations as we see in Tables II and IV. We report these results to assist other practitioners in the design of parametrized *Ansätze* for scientific applications.

We also report on the influence of hardware noise on the QCBM performance. In our study, the QCBM were trained in the absence of noise and certain trained models were deployed on near-term devices. In our results we observe that hardware noise flattens out the landscape. Additionally, we observe that the addition of hardware noise does not lead to an increase in correlation between the encoded variables.

Training in the presence of hardware noise (e.g. using local parameter tuning or LCD) can improve performance; however, without robust error mitigation hardware fluctuations can undo small improvements in performance (see Fig. 14). Designing scalable error mitigation methods that fully capture correlations in the hardware is an active area of research in quantum computing. For example, a commonly employed method of readout error mitigation is using measurement fidelity matrices [34–36]. These methods have an advantage in that they can be incorporated into variational training workflows (either gradient-based optimization or LCD) as a data postprocessing step. This motivates the need for a systematic follow-up study of error mitigation efficacy in this application, with a focus on balancing overhead with model performance.

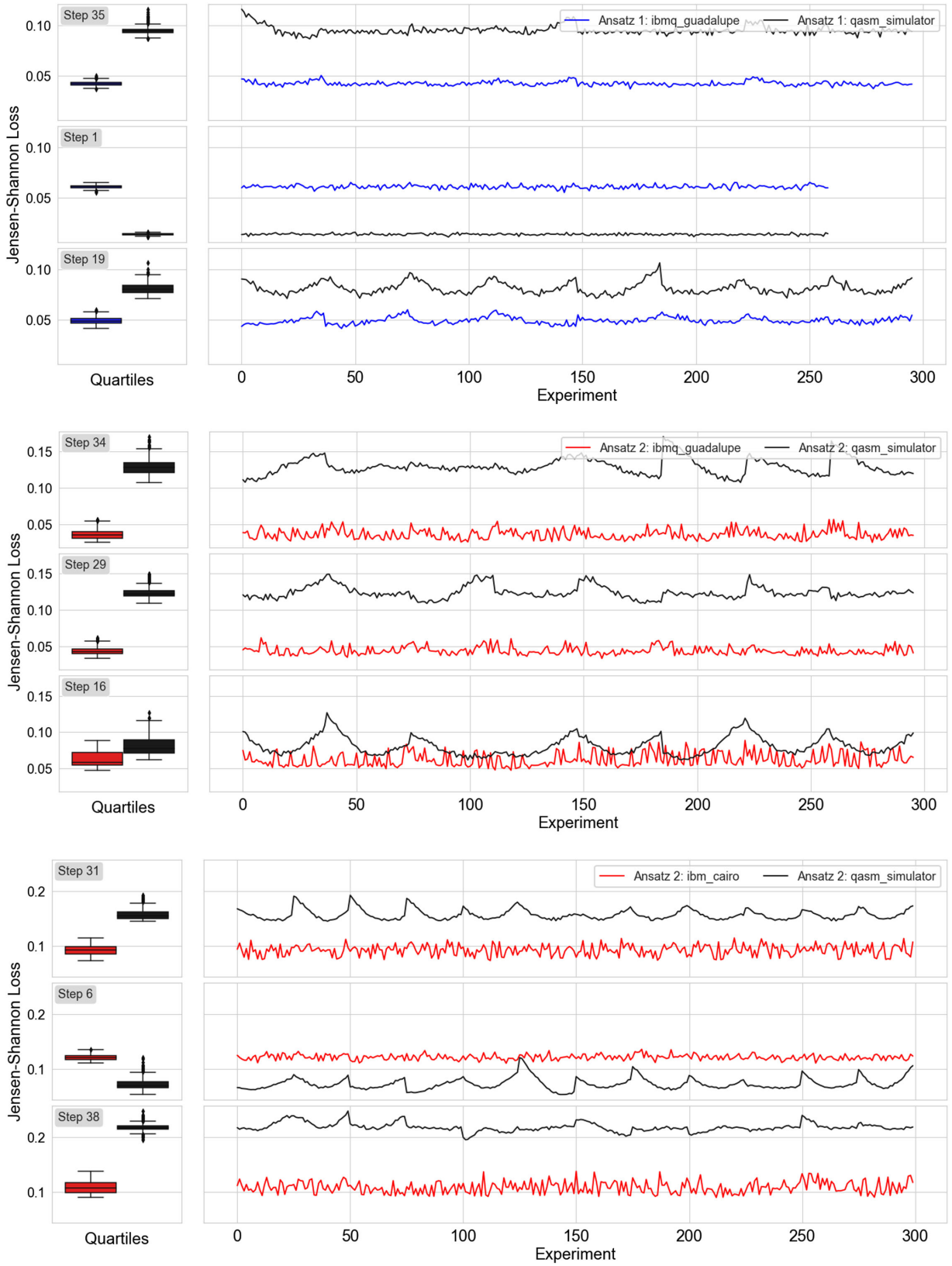

FIG. 15. Left: quartiles of Jensen-Shannon loss for three representative iterations in the LCD scheme. Each iteration (right) corresponds to a parameter sweep in the range $[-\frac{\pi}{4}, \frac{\pi}{4}]$ for a particular rotation gate in a given layer. The plots display the JS values for a eight-qubit QCBM trained on `ibmq_guadalupe` using *Ansatz* 1 (top) and *Ansatz* 2 (middle). The bottom plot corresponds to a 12-qubit QCBM constructed using *Ansatz* 2 and trained on `ibm_cairo`.

ACKNOWLEDGMENTS

This work was partially supported by the Quantum Information Science Enabled Discovery (QuantISED) for High Energy Physics program at ORNL under FWP ERKAP61. This work was partially supported by the Laboratory Directed Research and Development Program of ORNL, managed by UT-Battelle, LLC, for the U.S. Department of Energy. This work was partially supported as part of the ASCR Testbed Pathfinder Program at Oak Ridge National Laboratory under FWP ERKJ332. This work was partially supported as part of the ASCR Fundamental Algorithmic Research for Quantum Computing Program at Oak Ridge National Laboratory under FWP ERKJ354. This research used quantum computing system resources of the Oak Ridge Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract No. DE-AC05-00OR22725. Oak Ridge National Laboratory manages access to the IBM Q System as part of the IBM Q Network. The authors would like to thank Dr. Phil Lotshaw for helpful comments during the manuscript preparation. This work has been partially supported by U.S. DOE Grant No. DE-FG02-13ER41967. ORNL is managed by UT-Battelle, LLC, under Contract No. DE-AC05-00OR22725 for the U.S. Department of Energy. The U.S. Government retains and the publisher, by accepting the article for publication, acknowledges that the U.S. Government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this manuscript, or allow others to do so, for the U.S. Government purposes. The Department of Energy will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan.

[1] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio, Generative adversarial nets, in *Advances in Neural Information Processing Systems*, edited by Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger (NIPS, 2014), Vol. 27.

[2] Michela Paganini, Luke de Oliveira, and Benjamin Nachman, Calogan: Simulating 3d high energy particle showers in multilayer electromagnetic calorimeters with generative adversarial networks, Phys. Rev. D **97**, 014021 (2018).

[3] Luke de Oliveira, Michela Paganini, and Benjamin Nachman, Learning particle physics by example: Location-aware generative adversarial networks for physics synthesis, Comput. Software Big Sci. **1**, 4 (2017).

[4] Pasquale Musella and Francesco Pandolfi, Fast and accurate simulation of particle detectors using generative adversarial networks, Comput. Software Big Sci. **2**, 8 (2018).

[5] Anja Butter, Tilman Plehn, and Ramon Winterhalder, How to GAN LHC events, SciPost Phys. **7**, 075 (2019).

[6] Riccardo Di Sipio, Michele Faucci Giannelli, Sana Ketabchi Haghighat, and Serena Palazzo, Dijetgan: A generative-adversarial network approach for the simulation of QCD dijet events at the LHC, J. High Energy Phys. 08 (2019) 110.

[7] Bobak Hashemi, Nick Amin, Kaustuv Datta, Dominick Olivito, and Maurizio Pierini, LHC analysis-specific datasets with generative adversarial networks, arXiv:1901.05282.

[8] Yasir Alanazi, Nobuo Sato, Tianbo Liu, Wally Melnitchouk, Pawel Ambrozewicz, Florian Hauenstein, Michelle P. Kuchera, Evan Pritchard, Michael Robertson, Ryan Strauss, Luisa Velasco, and Yaohang Li, Simulation of electron-proton scattering events by a feature-augmented and transformed generative adversarial network (FAT-GAN), in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence* (International Joint Conferences on Artificial Intelligence Organization, 2021).

[9] Carlos Bravo-Prieto, Julien Baglio, Marco Cè, Anthony Francis, Dorota M. Grabowska, and Stefano Carrazza, Style-based quantum generative adversarial networks for Monte Carlo events, Quantum **6**, 777 (2022).

[10] Su Yeon Chang, Steven Herbert, Sofia Vallecorsa, Elías F. Combarro, and Ross Duncan, Dual-parametrized quantum circuit GAN model in high energy physics, EPJ Web Conf. **251**, 03050 (2021).

[11] Su Yeon Chang, Sofia Vallecorsa, Elías F. Combarro, and Federico Carminati, Quantum generative adversarial networks in a continuous-variable architecture to simulate high energy physics detectors, arXiv:2101.11132.

[12] Adrián Pérez-Salinas, Juan Cruz-Martinez, Abdulla A Alhajri, and Stefano Carrazza, Determining the proton content with a quantum computer, Phys. Rev. D **103**, 034027 (2021).

[13] Jin-Guo Liu and Lei Wang, Differentiable learning of quantum circuit Born machine, Phys. Rev. A **98**, 062324 (2018).

[14] Kathleen E. Hamilton, Eugene F. Dumitrescu, and Raphael C. Pooser, Generative model benchmarks for superconducting qubits, Phys. Rev. A **99**, 062323 (2019).

[15] Marcello Benedetti, Delfina Garcia-Pintos, Yunseong Nam, and Alejandro Perdomo-Ortiz, A generative modeling approach for benchmarking and training shallow quantum circuits, npj Quantum Inf. **5**, 45 (2019).

[16] Shakir Mohamed and Balaji Lakshminarayanan, Learning in implicit generative models, arXiv:1610.03483.

[17] Sukin Sim, Peter D. Johnson, and Alán Aspuru-Guzik, Expressibility and entangling capability of parametrized quantum circuits for hybrid quantum-classical algorithms, Adv. Quantum Technol. **2**, 1900070 (2019).

[18] Kouhei Nakaji and Naoki Yamamoto, Expressibility of the alternating layered ansatz for quantum computation, Quantum **5**, 434 (2021).

[19] Yuxuan Du, Min-Hsiu Hsieh, Tongliang Liu, and Dacheng Tao, Expressive power of parametrized quantum circuits, Phys. Rev. Res. **2**, 033125 (2020).
[20] Jarrod R McClean, Sergio Boixo, Vadim N Smelyanskiy, Ryan Babbush, and Hartmut Neven, Barren plateaus in quantum neural network training landscapes, Nat. Commun. **9**, 4812 (2018).
[21] M Cerezo, Akira Sone, Tyler Volkoff, Lukasz Cincio, and Patrick J Coles, Cost function dependent barren plateaus in shallow parametrized quantum circuits, Nat. Commun. **12**, 1791 (2021).
[22] Ville Bergholm, Josh Izaac, Maria Schuld, Christian Gogolin, M. Sohaib Alam, Shahnawaz Ahmed, Juan Miguel Arrazola, Carsten Blank, Alain Delgado, Soran Jahangiri, Keri McKiernan, Johannes Jakob Meyer, Zeyue Niu, Antal Száva, and Nathan Killoran, Pennylane: Automatic differentiation of hybrid quantum-classical computations, arXiv:1811.04968.
[23] Maria Schuld, Ville Bergholm, Christian Gogolin, Josh Izaac, and Nathan Killoran, Evaluating analytic gradients on quantum hardware, Phys. Rev. A **99**, 032331 (2019).
[24] Diederik P. Kingma and Jimmy Ba, ADAM: A method for stochastic optimization, arXiv:1412.6980.
[25] Johan Alwall, Michel Herquet, Fabio Maltoni, Olivier Mattelaer, and Tim Stelzer, MADGRAPH 5: Going beyond, J. High Energy Phys. 06 (2011) 128.
[26] Torbjörn Sjöstrand, Stephen Mrenna, and Peter Skands, A brief introduction to PYTHIA8.1, Comput. Phys. Commun. **178**, 852 (2008).
[27] J. de Favereau, C. Delaere, P. Demin, A. Giammanco, V. Lemaître, A. Mertens, and M. Selvaggi, DELPHES 3: A modular framework for fast simulation of a generic collider experiment, J. High Energy Phys. 02 (2014) 057.
[28] Matteo Cacciari, Gavin P Salam, and Gregory Soyez, The anti-k_t jet clustering algorithm, J. High Energy Phys. 04 (2008) 063.
[29] Matteo Cacciari, FASTJET: A code for fast k_t clustering, and more, arXiv:hep-ph/0607071.
[30] Sebastian Bieringer, Anja Butter, Sascha Diefenbacher, Engin Eren, Frank Gaede, Daniel Hundhausen, Gregor Kasieczka, Benjamin Nachman, Tilman Plehn, and Mathias Trabs, Calomplification—The power of generative calorimeter models, J. Instrum. **17**, P09028 (2022).
[31] Anja Butter, Sascha Diefenbacher, Gregor Kasieczka, Benjamin Nachman, and Tilman Plehn, GANplifying event samples, SciPost Phys. **10**, 139 (2021).
[32] Samson Wang, Enrico Fontana, M. Cerezo, Kunal Sharma, Akira Sone, Lukasz Cincio, and Patrick J. Coles, Noise-induced barren plateaus in variational quantum algorithms, Nat. Commun. **12**, 6961 (2021).
[33] Zoë Holmes, Kunal Sharma, M Cerezo, and Patrick J Coles, Connecting ansatz expressibility to gradient magnitudes and barren plateaus, PRX Quantum **3**, 010313 (2022).
[34] Kathleen E. Hamilton and Raphael C. Pooser, Error-mitigated data-driven circuit learning on noisy quantum hardware, Quantum Mach. Intell. **2**, 10 (2020).
[35] Kathleen E. Hamilton, Tyler Kharazi, Titus Morris, Alexander J. McCaskey, Ryan S. Bennink, and Raphael C. Pooser, Scalable quantum processor noise characterization, in *Proceedings of the 2020 IEEE International Conference on Quantum Computing and Engineering (QCE)* (IEEE, New York, 2020), pp. 430–440.
[36] Sergey Bravyi, Sarah Sheldon, Abhinav Kandala, David C. Mckay, and Jay M. Gambetta, Mitigating measurement errors in multiqubit experiments, Phys. Rev. A **103**, 042605 (2021).