QAMP 2025

Enhancing Quantum Diffusion Models for Complex Image Generation

Jeongbin Jo[1], Santanam Wishal[2], Shah Md Khalil Ullah[3], and Shan Kowalski[4]

[1]Department of Physics , Yonsei University , *jeongbin033@yonsei.ac.kr*
[2]Asia Cyber University , *santawishal17@gmail.com*
[3]Khulna University of Engineering and Technology , *hamimkhandakar222@gmail.com*
[4]Eleven Dimension LLC , *shanjdk2012@gmail.com*

December 21, 2025

Contents

1 Intrduction **2**
1.1 What's the potential benefit of replacing neural network with PQCs? 2
1.2 How does the proposed QDM compare to just classical DM? 2

2 Diffusion Model **3**
2.1 Classical Diffusion Model 3
2.1.1 Forward Process: Diffusion Process 3
2.1.2 Reverse Process: Generative Process 3
2.1.3 Loss Function 4
2.1.4 Continuous-Time Formulation: SDE 4
2.2 Quantum Diffusion Model 4
2.2.1 Forward Process: Depolarizing Channel 5
2.2.2 Reverse Process: Quantum Denoising 5
2.2.3 Loss Function: Fidelity or Trace Distance 5

3 Quantum Data Encoding **5**
3.1 Basis Encoding 6
3.2 Amplitude Encoding 6
3.3 Angle Encoding 6
3.4 Phase Encoding 6
3.5 Dense Angle Encoding 6
3.6 Architecture 7

4 Experimental Results 8

5 Conclusion 9

---

1 Intrduction

1.1 What's the potential benefit of replacing neural network with PQCs?

Parameter Efficiency (Model Compression)

The authors state in their conclusion that "quantum diffusion models require fewer parameters with respect to their classical counterpart". For example, their 'conditioned latent' model, which generated all 10 MNIST digits, used only 1110 parameters. This is a massive reduction compared to the millions of parameters typical in classical U-Net architectures used for diffusion.

Expressivity and Scalability

The paper suggests that for a "full quantum case," it is "possible to generate distributions whose features scale exponentially with the number of qubits". This implies PQCs have a fundamentally greater expressive power.

Future Potential

The long-term benefit is the potential to "approximate probability distributions that are not classically tractable", including "quantum datasets" themselves.

1.2 How does the proposed QDM compare to just classical DM?

Hybrid Architecture

The paper's QDM is a hybrid model. The forward process (adding noise) is performed classically. The backward process (denoising) is performed by a PQC.

PQC as a Denoising Operator

Instead of having the PQC predict the noise ϵ_θ (like a classical U-Net), this model trains the PQC to be a direct denoising operator $P(\theta, t)$ such that $**P(\theta, t)|x_t\rangle = |x_{t-1}\rangle**$. This avoids the "impractical" cycle of decoding/encoding quantum states at every timestep.

Use of Complex Noise

Because PQC operations can make state coefficients complex, the authors found that using complex Gaussian noise ($\epsilon = \epsilon_r + i\epsilon_i$) during the classical forward process led to a "significant improvement in performance". This is a notable departure from classical models.

Comparable Performance (in Latent Space)

The paper's 'Latent QDM' (which uses a classical autoencoder first) achieved a competitive FID of 38.2 and was able to generate "samples of similar quality to those generated by classical algorithms".

2 Diffusion Model

2.1 Classical Diffusion Model

Diffusion models are a class of generative models in machine learning inspired by non-equilibrium thermodynamics. The goal is to learn a diffusion process that destroys structure in data, and then learn a reverse process that restores structure to generate new data samples from noise.[3]

2.1.1 Forward Process: Diffusion Process

The forward process is defined by a Markov chain that gradually adds Gaussian noise to the data according to a variance schedule $\beta_1, \ldots, \beta_T$.

$$q(x_t|x_{t-1}) = \mathcal{N}\left(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t \mathbf{I}\right) \tag{1}$$

Using the notation $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$, we can sample x_t at any arbitrary time step t directly from x_0 in closed form:

$$q(x_t|x_0) = \mathcal{N}\left(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)\mathbf{I}\right) \tag{2}$$

This property allows for efficient training by eliminating the need to iterate through all previous timesteps.

2.1.2 Reverse Process: Generative Process

The generative process[4] is defined as the reverse Markov chain. Since the exact reverse posterior $q(x_{t-1}|x_t)$ is intractable, we approximate it using a neural network with parameters θ:

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}\left(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)\right) \tag{3}$$

Here, $\mu_\theta(x_t, t)$ is the predicted mean, and $\Sigma_\theta(x_t, t)$ is the covariance (often fixed to $\beta_t \mathbf{I}$ or learned).

2.1.3 Loss Function

Training is performed by optimizing the variational lower bound (VLB) on the negative log-likelihood.

$$\mathcal{L} = \mathbb{E}\left[-\log p_\theta(\mathbf{x}_0)\right] \leq \mathbb{E}_q\left[-\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] \tag{4}$$

This can be rewritten as a sum of KL divergence[7] terms:

$$\mathcal{L} = \mathbb{E}_q\left[\underbrace{D_{KL}(q(\mathbf{x}_T|\mathbf{x}_0)||p(\mathbf{x}_T))}_{L_T} + \sum_{t>1}\underbrace{D_{KL}(q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)||p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))}_{L_{t-1}}\underbrace{-\log p_\theta(\mathbf{x}_0|\mathbf{x}_1)}_{L_0}\right] \tag{5}$$

In practice, Ho et al. found that a simplified objective function yields better sample quality. The objective is to predict the noise ϵ added to x_0:

$$\mathcal{L}_{\text{simple}}(\theta) = \mathbb{E}_{t,x_0,\epsilon}\left[\|\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, t)\|^2\right] \tag{6}$$

where $\epsilon \sim \mathcal{N}(0, \mathbf{I})$ and ϵ_θ is a function approximator (e.g., U-Net).

2.1.4 Continuous-Time Formulation: SDE

The discrete diffusion process can be generalized to continuous time using Stochastic Differential Equations (SDEs). The forward process is described by the following Itô SDE[6]:

$$d\mathbf{x} = \mathbf{f}(\mathbf{x}, t)dt + g(t)d\mathbf{w} \tag{7}$$

where $\mathbf{f}(\cdot, t)$ is the drift coefficient, $g(t)$ is the diffusion coefficient, and $\mathbf{w}$ is a standard Wiener process.

Remarkably, the reverse process—generating data from noise—is also a diffusion process governed by the *reverse-time SDE*[1]:

$$d\mathbf{x} = \left[\mathbf{f}(\mathbf{x}, t) - g(t)^2\nabla_{\mathbf{x}}\log p_t(\mathbf{x})\right]dt + g(t)d\bar{\mathbf{w}} \tag{8}$$

Here, dt represents a negative time step, and $\bar{\mathbf{w}}$ is the Brownian motion in reverse time. The term $\nabla_{\mathbf{x}}\log p_t(\mathbf{x})$, known as the *score function*, points towards high-density regions of the data.

Therefore, the core task of the diffusion model is to learn a score-based model $s_\theta(\mathbf{x}, t) \approx \nabla_{\mathbf{x}}\log p_t(\mathbf{x})$ using a neural network (or PQC in our case), allowing us to numerically solve the reverse SDE to generate samples from noise.

2.2 Quantum Diffusion Model

While classical diffusion models operate on probability distributions over classical data, Quantum Diffusion Models (QDMs) extend this concept to the Hilbert space of quantum states. The goal is to generate quantum states (density matrices) from a maximally mixed state.

2.2.1 Forward Process: Depolarizing Channel

Instead of adding Gaussian noise, the forward process in QDM is typically modeled as a standard depolarizing channel acting on a density matrix ρ. For a system of n qubits with dimension $d = 2^n$, the state at step t is given by:

$$\rho_t = \mathcal{E}_t(\rho_{t-1}) = (1 - p_t)\rho_{t-1} + p_t \frac{I}{d} \tag{9}$$

where $p_t \in [0, 1]$ is the depolarization probability (noise schedule). Similar to the classical case, we can express ρ_t directly from the initial state ρ_0. Let $\alpha_t = \prod_{s=1}^{t}(1 - p_s)$, then:

$$\rho_t = \alpha_t \rho_0 + (1 - \alpha_t)\frac{I}{d} \tag{10}$$

As $t \to T$, $\alpha_T \to 0$, and the state converges to the maximally mixed state $\rho_T \approx \frac{I}{d}$, which contains no information about ρ_0.

2.2.2 Reverse Process: Quantum Denoising

The reverse process aims to restore the quantum state from the noise. This is modeled by a parameterized quantum circuit (PQC), denoted as a unitary operator $U(\theta)$. The discrete reverse step can be approximated as:

$$\rho_{t-1} \approx \mathcal{D}_\theta(\rho_t, t) = U(\theta_t)\rho_t U^\dagger(\theta_t) \tag{11}$$

For more complex generative tasks, the reverse process may involve ancillary qubits and measurements to simulate non-unitary maps.

2.2.3 Loss Function: Fidelity or Trace Distance

Unlike the KL divergence used in classical models, quantum models often use Fidelity or Hilbert-Schmidt distance to measure the closeness between the generated state and the target state. A common objective is to maximize the overlap with the target pure state $|\psi\rangle$:

$$\mathcal{L}(\theta) = 1 - \mathbb{E}\left[\langle\psi| \rho_{\text{gen}}(\theta) |\psi\rangle\right] \tag{12}$$

Alternatively, for density matrices, we minimize the trace distance or maximize the quantum fidelity $F(\rho, \sigma) = (\text{tr}\sqrt{\sqrt{\rho}\sigma\sqrt{\rho}})^2$.

3 Quantum Data Encoding

We referred to IBM Quantum Platform, especially Quantum Machine Learning.[5]

3.1 Basis Encoding

Basis encoding maps a classical P-bit string directly to a computational basis state of a P-qubit system. For a single feature represented as binary bits $(b_1, b_2, \ldots, b_P)$, the quantum state is:

$$|x\rangle = |b_1, b_2, \ldots, b_P\rangle, \quad b_i \in \{0, 1\} \tag{13}$$

3.2 Amplitude Encoding

Amplitude encoding maps a normalized classical N-dimensional data vector $\vec{x}$ to the amplitudes of an n-qubit quantum state, where $n = \lceil \log_2 N \rceil$.

$$|\psi_x\rangle = \frac{1}{\alpha} \sum_{i=1}^{N} x_i |i\rangle \tag{14}$$

Here, $|i\rangle$ is the computational basis state and α is a normalization constant ensuring $\langle \psi_x | \psi_x \rangle = 1$. And α is a normalization constant to be determined from the data being encoded.

$$\alpha = \sqrt{\sum_{i=1}^{N} |x_i|^2} \tag{15}$$

3.3 Angle Encoding

Angle encoding maps each feature x_k to the rotation angle of a qubit using R_Y gates. For a data vector $\vec{x}$, the state is a product state:

$$|\vec{x}\rangle = \bigotimes_{k=1}^{N} R_Y(x_k) |0\rangle = \bigotimes_{k=1}^{N} \left(\cos\left(\frac{x_k}{2}\right) |0\rangle + \sin\left(\frac{x_k}{2}\right) |1\rangle \right) \tag{16}$$

3.4 Phase Encoding

Phase encoding maps data features to the phase of qubits using Phase gates $P(\phi)$, typically applied after Hadamard gates.

$$|\vec{x}\rangle = \bigotimes_{k=1}^{N} P(x_k) |+\rangle = \frac{1}{\sqrt{2^N}} \bigotimes_{k=1}^{N} \left(|0\rangle + e^{ix_k} |1\rangle \right) \tag{17}$$

3.5 Dense Angle Encoding

Dense angle encoding encodes two features, x_k and x_l, into a single qubit using both Y-axis and Z-axis rotations.

$$|x_k, x_l\rangle = R_Z(x_l) R_Y(x_k) |0\rangle = \cos\left(\frac{x_k}{2}\right) |0\rangle + e^{ix_l} \sin\left(\frac{x_k}{2}\right) |1\rangle \tag{18}$$

Extending this to more features, the data vector $\vec{x} = (x_1, ..., x_N)$ can be encoded as:

$$|\vec{x}\rangle = \bigotimes_{k=1}^{N/2} \left(\cos x_{2k-1} |0\rangle + e^{ix_{2k}} \sin x_{2k-1} |1\rangle\right) \tag{19}$$

3.6 Architecture

The proposed model, illustrated in Figure 1, adopts a hybrid quantum-classical U-Net architecture inspired by recent advances in quantum generative diffusion models[2, 8]. The architecture is designed to leverage the expressivity of quantum circuits while maintaining the reconstruction capability of classical networks. It consists of three distinct modules: a classical encoder, a quantum bottleneck (core), and a classical decoder.

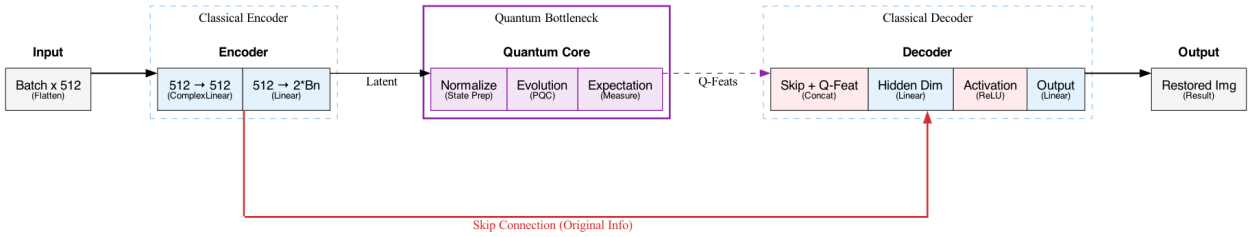


Figure 1: Schematic of the Hybrid Quantum-Classical U-Net Architecture. The model integrates a quantum bottleneck for feature extraction and employs a skip connection to preserve original semantic information during the reverse diffusion process.

First, the **Classical Encoder** transforms the input data (flattened vector) into a lower-dimensional latent representation. To seamlessly interface with the quantum layer, we utilize complex-valued linear layers (`ComplexLinear`) that preserve phase information necessary for quantum state preparation.

Second, the **Quantum Bottleneck** acts as the core generative component. We employ *amplitude encoding* to efficiently map the classical latent vector $\mathbf{z}$ onto an n-qubit quantum state $|\psi\rangle$ by normalizing the complex vector. This state is evolved by a Parameterized Quantum Circuit (PQC), $U(\theta)$, which learns the reverse diffusion dynamics. The resulting quantum state is then measured to extract quantum features (Q-Feats).

Finally, the **Classical Decoder** reconstructs the denoised sample from the quantum features. A crucial design element is the inclusion of a **Skip Connection** (red path in Figure 1). This connection concatenates the original flattened input vector directly with the quantum features in the decoder. This mechanism is essential for mitigating the "bottleneck" problem inherent in low-qubit quantum circuits and preventing the model from converging to trivial identity mappings by preserving the original signal structure.

4 Experimental Results

We evaluated the proposed Hybrid Quantum U-Net using the MNIST dataset. The training was conducted for 100 epochs, and the model demonstrated stable convergence behavior. As shown in Figure 2, the final infidelity loss reached approximately 0.25. Remarkably, the total execution time was only about 3 minutes, highlighting the computational efficiency of our hybrid architecture compared to fully classical counterparts or more complex quantum simulations.

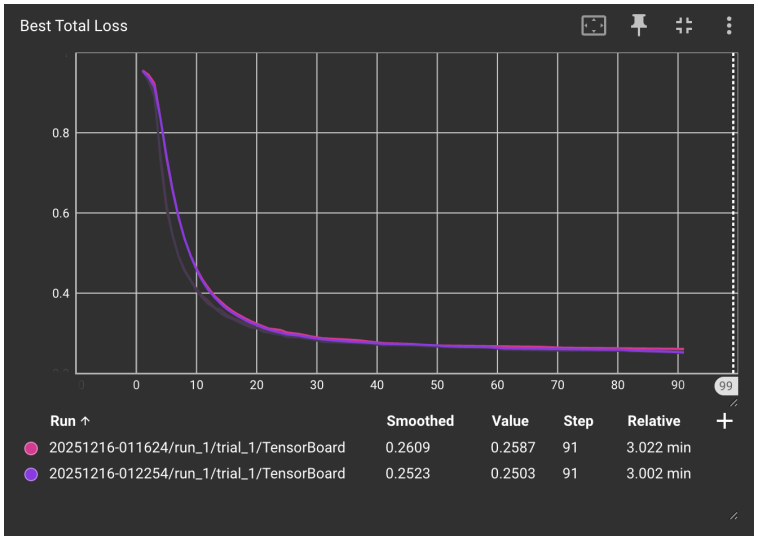


Figure 2: Training Loss Convergence. The model achieves a stable loss of ≈ 0.25 within 100 epochs, demonstrating rapid convergence.

Figure 3 visualizes the forward diffusion process used in our experiment. The input data is gradually corrupted by adding Gaussian noise over discrete time steps t, eventually converging to an isotropic Gaussian distribution.

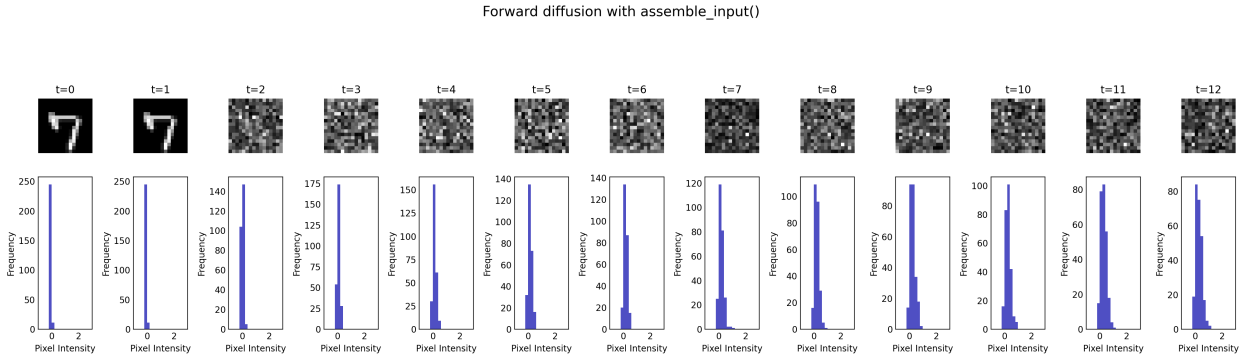
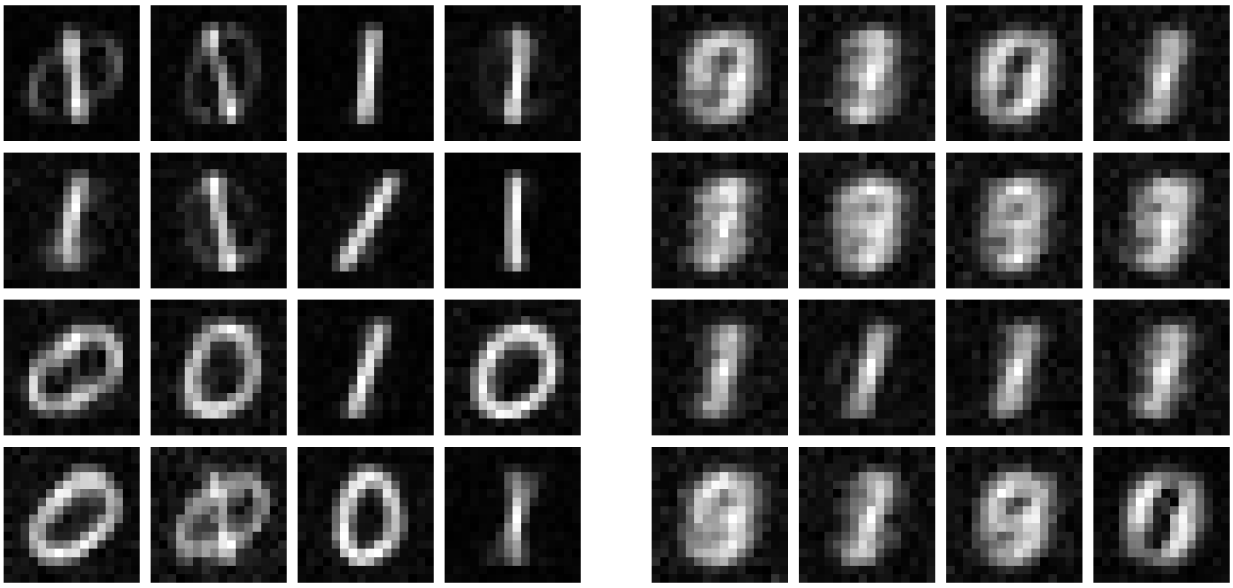


Figure 3: Visualization of the Forward Diffusion Process. The original digits (left) are progressively transformed into random noise (right).

To analyze the generative capability and scalability of the model, we compared two scenarios: a low-complexity subset (digits 0 and 1) and the full dataset (digits 0 through 9). The generation results are presented in Figure 4.



(a) Binary Class (Digits 0, 1)

(b) Multi-Class (Digits 0-9)

Figure 4: Comparison of Generated Samples. (a) shows sharp features for the binary case, while (b) exhibits increased ambiguity and blurring due to higher data complexity.

As observed in Figure 4a, the model successfully generates distinct and sharp images when trained on binary classes. However, as the complexity increases to include all ten digits (Figure 4b), the generated samples exhibit ambiguous shapes and blurring. This suggests that while the current quantum bottleneck is efficient, its limited Hilbert space (number of qubits) constrains the expressibility required to fully capture the complex multimodal distribution of the entire dataset.

5 Conclusion

In this study, we proposed a **Hybrid Quantum-Classical U-Net** architecture for complex image generation and validated its effectiveness using the MNIST dataset. The experimental results demonstrated that the proposed model exhibits significant generative capabilities with a reduced number of parameters, confirming the potential of hybrid structures in the NISQ era. The key insights gained, technical challenges faced, and our current research direction are summarized below.

Key Insights and Learnings

- **Structural Efficiency:** Through code implementation (`QuantumUNet`), we verified that the 'Quantum Bottleneck' architecture—which compresses features via a classical encoder before mapping them to a quantum state is significantly more efficient than encoding the entire high-dimensional image directly into qubits.

- **Importance of Skip Connections:** We discovered that **Skip Connections**, which combine original classical features with quantum features in the decoder stage, are critical. Without them, the information loss inherent in the quantum circuit significantly degrades image sharpness. This highlights that preserving the **Classical Path** is essential for maintaining reconstruction quality in hybrid models.

Technical Challenges

- **Simulation Overhead (Running Speed):** A major practical barrier was the **ML running speed**. Simulating quantum circuits (especially tensor product operations for entanglement) on classical hardware incurs massive computational costs. This overhead severely constrained our ability to scale up the number of qubits and circuit depth within a reasonable training time.

- **Barren Plateaus & Optimization Landscape:** Beyond computational speed, we faced the fundamental challenge of **Barren Plateaus**. The standard Adam optimizer struggled to navigate the flat loss landscapes typical of parameterized quantum circuits (PQCs). This often led to stagnation or convergence to local minima, particularly when attempting to learn the complex multi-modal distributions of the multi-class MNIST dataset.

Current Status: Enhancing Optimization with QNGD

To address these optimization challenges and ensure convergence stability, we are currently implementing **Quantum Natural Gradient Descent (QNGD)**.

- **Addressing the Optimization Geometry:** Standard optimizers like Adam assume a Euclidean parameter space, which is ill-suited for the quantum state space that follows Riemannian geometry. We are adopting QNGD to adjust the gradient direction by incorporating the **Fubini-Study Metric Tensor** (g_{ij}), which represents the curvature of the quantum information geometry. The update rule is defined as:

$$\theta_{t+1} = \theta_t - \eta g^{-1}(\theta_t) \nabla \mathcal{L}(\theta_t) \tag{20}$$

  where η is the learning rate and $g^{-1}(\theta_t)$ corrects the gradient based on the local curvature.

- **Quality over Speed (Trade-off):** We acknowledge that calculating the metric tensor increases the computational cost per step, potentially lengthening the wall-clock training time. However, we accept this trade-off to **mitigate Barren Plateaus** and achieve higher **Fidelity** with fewer training epochs. Our focus has shifted from raw simulation speed to **'Optimization Efficiency'**, aiming to overcome the expressivity and convergence limitations of the current model.

References

[1] Brian D.O. Anderson. "Reverse-time diffusion equation models". In: *Stochastic Processes and their Applications* 12.3 (1982), pp. 313–326. ISSN: 0304-4149. DOI: https://doi.org/10.1016/0304-4149(82)90051-5. URL: https://www.sciencedirect.com/science/article/pii/0304414982900515.

[2] Andrea Cacioppo et al. *Quantum Diffusion Models*. 2023. arXiv: 2311.15444 [quant-ph]. URL: https://arxiv.org/abs/2311.15444.

[3] *Diffusion Model*. URL: https://en.wikipedia.org/wiki/Diffusion_model (visited on 12/07/2025).

[4] Jonathan Ho, Ajay Jain, and Pieter Abbeel. "Denoising Diffusion Probabilistic Models". In: *CoRR* abs/2006.11239 (2020). arXiv: 2006.11239. URL: https://arxiv.org/abs/2006.11239.

[5] *IBM Quantum Platform Quantum Machine Learning*. URL: https://quantum.cloud.ibm.com/learning/en/courses/quantum-machine-learning/data-encoding#methods-of-encoding (visited on 12/13/2025).

[6] *Ito calculus*. URL: https://en.wikipedia.org/wiki/It%C3%B4_calculus (visited on 12/07/2025).

[7] *Kullback-Leibler Divergence*. URL: https://en.wikipedia.org/wiki/Kullback%E2%80%93Leibler_divergence (visited on 12/07/2025).

[8] *Quantum Diffusion Model*. URL: https://github.com/AndreaCacioppo/quantum-diffusion-models-code (visited on 12/13/2025).