entropy

MDPI

Article

QEKI: A Quantum–Classical Framework for Efficient Bayesian Inversion of PDEs

Jiawei Yong and Sihai Tang *

School of Information Science and Technology, ShanghaiTech University, Shanghai 201210, China; yongjw2023@shanghaitech.edu.cn

* Correspondence: tangsh2023@shanghaitech.edu.cn

Abstract

Solving Bayesian inverse problems efficiently stands as a major bottleneck in scientific computing. Although Bayesian Physics-Informed Neural Networks (B-PINNs) have introduced a robust way to quantify uncertainty, the high-dimensional parameter spaces inherent in deep learning often lead to prohibitive sampling costs. Addressing this, our work introduces Quantum-Encodable Bayesian PINNs trained via Classical Ensemble Kalman Inversion (QEKI), a framework that pairs Quantum Neural Networks (QNNs) with Ensemble Kalman Inversion (EKI). The core advantage lies in the QNN's ability to act as a compact surrogate for PDE solutions, capturing complex physics with significantly fewer parameters than classical networks. By adopting the gradient-free EKI for training, we mitigate the barren plateau issue that plagues quantum optimization. Through several benchmarks on 1D and 2D nonlinear PDEs, we show that QEKI yields precise inversions and substantial parameter compression, even in the presence of noise. While large-scale applications are constrained by current quantum hardware, this research outlines a viable hybrid framework for including quantum features within Bayesian uncertainty quantification.

Keywords: physics-informed neural networks; quantum neural networks; ensemble Kalman inversion; noisy data

1. Introduction

Partial Differential Equations (PDEs) are important tools for describing the dynamics of physical, chemical, and engineering systems. In addition to state variables, these models often depend on multiple parameters that characterize material properties, source terms, or boundary conditions. In many real-world scenarios, some of these parameters are often unknown and must be inferred from limited and noisy measurement data, which are often referred to as PDE inverse problems.

Physics-Informed Neural Networks (PINNs) [1,2] provide a flexible framework for solving partial differential equations by embedding the governing physical laws into the loss function. This formulation has been widely used in both forward and inverse PDE problems. In parallel, operator learning methods have been developed to approximate mappings between function spaces. DeepONet [3] adopts a dual-network architecture, in which a branch network encodes input functions and a trunk network represents spatial coordinates, enabling the approximation of nonlinear operators. Another representative approach is the Fourier Neural Operator (FNO) [4], which parameterizes integral kernels in the frequency domain using Fourier representations and has been applied to a range of problems involving complex physical systems.

Check for updates

Academic Editor: Younghun Kwon

Received: 12 December 2025
Revised: 25 January 2026
Accepted: 28 January 2026
Published: 30 January 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

To further enable uncertainty quantification and Bayesian inference, the Bayesian Physics-Informed Neural Network (B-PINNs) framework [5] extends PINNs by treating network parameters probabilistically. B-PINNs originally rely on Hamiltonian Monte Carlo (HMC), which provides asymptotically exact posterior samples but is often computationally expensive. More recent work [6,7] instead adopts the Ensemble Kalman Inversion (EKI) [8,9], a gradient-free and efficient alternative that approximates posterior updates through ensemble-based dynamics.

The rise of Quantum Neural Networks (QNNs) [10,11] has brought new opportunities for deep learning and scientific computing. QNNs achieve a nonlinear representation and probabilistic modeling in a high-dimensional Hilbert space through a quantum feature mapping and variational quantum circuits. The strength of QNNs comes from their ability to combine quantum superposition with the intrinsic parallelism of neural networks, enabling a computational framework with significant potential [12,13]. It has been demonstrated in Ref. [14] that quantum models capable of realizing all sets of Fourier coefficients can act as simulators of universal functions. Several recent studies [15–17] have begun to explore integrating QNNs with Physics-Informed Neural Networks. However, training QNNs with gradient-based methods can suffer from the barren plateau phenomenon [18,19], where gradients vanish exponentially with the number of qubits or the depth of the circuit, posing challenges for optimization. Although previous research [20–22] has yielded a range of effective algorithms to mitigate this phenomenon, the quest for ever-improving, highly effective solutions remains of paramount importance.

Motivated by these developments, this paper proposes a Quantum-Encodable Bayesian PINNs trained via Classical Ensemble Kalman Inversion (QEKI) framework, which integrates QNNs into the traditional B-PINNs structure. Notably, the inherent structure of QNNs reduces the number of trainable parameters, which improves the representational efficiency of the inverse problem, allowing the architecture to achieve accurate posterior inference with fewer parameters. Furthermore, the EKI-based parameter update does not require gradient computation, which not only accelerates QNN training but also helps mitigate the barren plateau phenomenon. Although current experiments rely on simulated quantum devices, the accuracy and efficiency of the method are expected to improve further with the deployment of future quantum hardware.

The main contributions of this paper are summarized as follows:

(1) We propose a hybrid Quantum-Encodable Bayesian PINNs trained via Classical Ensemble Kalman Inversion (QEKI) framework that combines QNN-based agent modeling with EKI to achieve gradient-free training.
(2) We reduce the trainable parameters by more than an order of magnitude while achieving comparable accuracy. In addition, the method effectively avoids barren plateau behavior, offering a practical optimization method for QNNs.
(3) We provide numerical evidence on nonlinear Poisson, diffusion–reaction, and Burgers equations showing superior robustness to observation noise.

The remainder of the paper is organized as follows. In Section 2, we introduce the problem formulation, including the B-PINNs, HMC, EKI, and the principles of QNNs, and then present the algorithm of the proposed QEKI method. In Section 3, we describe the experimental setups and present the corresponding results for three representative test cases. Finally, in Section 4, we provide concluding remarks and discuss directions for future research.

2. Methodology

The problem we considered is formulated as follows.

$$\begin{aligned} \mathcal{N}_{\boldsymbol{x}}(u(\boldsymbol{x});\boldsymbol{\lambda}) &= f(\boldsymbol{x}), \quad \boldsymbol{x} \in D, \\ \mathcal{B}_{\boldsymbol{x}}(u(\boldsymbol{x});\boldsymbol{\lambda}) &= b(\boldsymbol{x}), \quad \boldsymbol{x} \in \partial D, \end{aligned} \tag{1}$$

where $\mathcal{N}_{\boldsymbol{x}}$ is the differential operator that describes the physical process, and $\mathcal{B}_{\boldsymbol{x}}$ is the boundary operator applied to the boundary. $D \subseteq \mathbb{R}^d$ is the d-dimensional physical domain with boundary ∂D, and $\boldsymbol{\lambda} \in \mathbb{R}^{N_\lambda}$ represents a vector of unknown physical parameters. In addition, $f(\boldsymbol{x})$ is the forcing function, $b(\boldsymbol{x})$ is the boundary function, and $u(\boldsymbol{x})$ is the solution of the PDE. The aim of the inverse problem is to infer the solution function $u(\boldsymbol{x})$ and the unknown parameters $\boldsymbol{\lambda}$ by integrating observational data with physical equations. The available data consists of the following three sets:

$$\mathcal{D}_f = \left\{\boldsymbol{x}_f^i, f^i\right\}_{i=1}^{N_f}, \ \mathcal{D}_b = \left\{\boldsymbol{x}_b^i, b^i\right\}_{i=1}^{N_b}, \ \mathcal{D}_u = \left\{\boldsymbol{x}_u^i, u^i\right\}_{i=1}^{N_u}, \tag{2}$$

where $\boldsymbol{x}_f, \boldsymbol{x}_u \in D$ and $\boldsymbol{x}_b \in \partial D$, f_i, b_i, u_i correspond to residual data, boundary-condition data and solution measurement in points $\boldsymbol{x}_f^i, \boldsymbol{x}_b^i, \boldsymbol{x}_u^i$, respectively, N_f, N_b, N_u denote the number of residual, boundary and solution data points.

2.1. Bayesian Physics-Informed Neural Network

The Bayesian Physics-Informed Neural Network (B-PINNs) employs a fully connected neural network as a surrogate model $\tilde{u}(\boldsymbol{x};\boldsymbol{\theta})$ for the PDE solution $u(\boldsymbol{x})$, where $\boldsymbol{\theta} \in \mathbb{R}^{N_\theta}$ represents the parameters of the neural network. Set $\boldsymbol{\xi} = \{\boldsymbol{\theta}, \boldsymbol{\lambda}\}$ collect both the surrogate model parameters and the unknown physical parameters. By Bayes' theorem, the posterior distribution of $\boldsymbol{\xi}$ is conditioned on the solution measurements D_u, residual data D_f, and boundary data D_b, which can be obtained as follows:

$$p(\boldsymbol{\xi} \mid \mathcal{D}_u, \mathcal{D}_f, \mathcal{D}_b) \propto p(\boldsymbol{\xi}) p(\mathcal{D}_u \mid \boldsymbol{\xi}) p(\mathcal{D}_f \mid \boldsymbol{\xi}) p(\mathcal{D}_b \mid \boldsymbol{\xi}). \tag{3}$$

We assume that the parameters are independent and that $\boldsymbol{\theta}$ follows a Gaussian distribution with zero mean . Therefore, we can express the prior as follows:

$$p(\boldsymbol{\xi}) = p(\boldsymbol{\lambda}) p(\boldsymbol{\theta}) = p(\boldsymbol{\lambda}) \prod_{i=1}^{N_\theta} p\left(\theta^i\right), \quad p\left(\theta^i\right) \sim \mathcal{N}\left(0, \sigma_\theta^{i\,2}\right), \tag{4}$$

where σ_θ^i is the standard deviation of the corresponding neural network parameter θ_i. In addition, the likelihood can be expressed as follows:

$$\begin{aligned} p(\mathcal{D}_u \mid \boldsymbol{\xi}) &= \prod_{i=1}^{N_u} p\left(u^i \mid \boldsymbol{\xi}\right), \quad p\left(\mathcal{D}_f \mid \boldsymbol{\xi}\right) = \prod_{i=1}^{N_f} p\left(f^i \mid \boldsymbol{\xi}\right), \quad p(\mathcal{D}_b \mid \boldsymbol{\xi}) = \prod_{i=1}^{N_b} p\left(b^i \mid \boldsymbol{\xi}\right), \\ p\left(u^i \mid \boldsymbol{\xi}\right) &= \frac{1}{\sqrt{2\pi\sigma_u^2}} \exp\left(-\frac{\left(u^i - \tilde{u}(\boldsymbol{x}_u^i;\boldsymbol{\theta})\right)^2}{2\sigma_u^2}\right), \\ p\left(f^i \mid \boldsymbol{\xi}\right) &= \frac{1}{\sqrt{2\pi\sigma_f^2}} \exp\left(-\frac{\left(f^i - \mathcal{N}_x\left(\tilde{u}\left(\boldsymbol{x}_f^i;\boldsymbol{\theta}\right);\boldsymbol{\lambda}\right)\right)^2}{2\sigma_f^2}\right), \\ p\left(b^i \mid \boldsymbol{\xi}\right) &= \frac{1}{\sqrt{2\pi\sigma_b^2}} \exp\left(-\frac{\left(b^i - \mathcal{B}_x(\tilde{u}(\boldsymbol{x}_b^i;\boldsymbol{\theta});\boldsymbol{\lambda})\right)^2}{2\sigma_b^2}\right), \end{aligned} \tag{5}$$

where $\sigma_u, \sigma_f, \sigma_b$. The prior distribution of the physical parameters $\boldsymbol{\lambda}$ is problem-dependent. After obtaining the prior and likelihood functions, various sampling methods can be used to derive the posterior distribution. Within the B-PINNs framework, efficient posterior sampling is essential for parameter estimation. HMC enables thorough posterior exploration, while EKI provides a gradient-free iterative update. Excessive trainable parameters may reduce sampling accuracy and increase computational cost. QNNs can serve as a partial remedy by lowering the number of parameters, potentially improving sampling efficiency and estimation quality.

2.2. Hamiltonian Monte Carlo

Hamiltonian Monte Carlo (HMC) is an efficient Markov Chain Monte Carlo (MCMC) method specifically designed for sampling from complex high-dimensional probability distributions, and has been applied to inverse problems in Bayesian Neural Networks. By introducing auxiliary momentum variables and simulating Hamiltonian dynamics, HMC is capable of generating states that are widely separated while maintaining a high acceptance probability.

Suppose that the target posterior distribution of $\boldsymbol{\xi}$ conditioned on the observations $\mathcal{D}_u, \mathcal{D}_f, \mathcal{D}_b$ is given by

$$p(\boldsymbol{\xi} \mid \mathcal{D}_u, \mathcal{D}_f, \mathcal{D}_b) \simeq \exp(-V(\boldsymbol{\xi})), \tag{6}$$

where $V(\boldsymbol{\xi}) = -\ln p(\mathcal{D}_u, \mathcal{D}_f, \mathcal{D}_b|\boldsymbol{\xi}) - \ln p(\boldsymbol{\xi})$ represents the potential energy. Then the Hamiltonian dynamics can be defined as follows:

$$H(\boldsymbol{\xi}, \boldsymbol{r}) = V(\boldsymbol{\xi}) + \frac{1}{2}\boldsymbol{r}^T \boldsymbol{M}^{-1}\boldsymbol{r}, \tag{7}$$

where $\boldsymbol{r}$ is an auxiliary momentum vector, and $\boldsymbol{M}$ is the corresponding mass matrix, which is set to be the identity matrix $\boldsymbol{I}$. $\frac{1}{2}\boldsymbol{r}^T \boldsymbol{M}^{-1}\boldsymbol{r}$ represents the kinetic energy. The Hamiltonian dynamics evolve the system according to

$$\begin{aligned} d\boldsymbol{\xi} &= -M^{-1}\boldsymbol{r}dt, \\ d\boldsymbol{r} &= -\nabla V(\boldsymbol{\xi})dt. \end{aligned} \tag{8}$$

We use the Leapfrog integration to perform updating, and the Metropolis–Hastings acceptance test determines whether the proposed sample is accepted. The complete HMC procedure is summarized in Algorithm 1.

2.3. Ensemble Kalman Inversion

Ensemble Kalman Inversion (EKI) is a gradient-free inversion method based on an ensemble of samples, originally developed from the Ensemble Kalman Filter (EnKF). Unlike traditional Bayesian inverse problem solutions, EKI does not require explicit gradient computation. Instead, it iteratively updates an ensemble of samples under observation constraints, progressively approximating the high-probability region of the posterior distribution.

For an inverse problem with known observation data y_{obs} and a forward model $\mathcal{G}$, the goal is to estimate the unknown parameters $\boldsymbol{\xi}$:

$$\boldsymbol{y}_{\text{obs}} = \mathcal{G}(\boldsymbol{\xi}) + \boldsymbol{\eta}, \tag{9}$$

where $\boldsymbol{\eta} \sim \mathcal{N}(0, R)$ is the observation noise, and R is the observation covariance matrix. According to Bayes' theorem, the posterior can be expressed in the following form:

$$p(\boldsymbol{\xi} \mid \boldsymbol{y}_{\text{obs}}) \propto p(\boldsymbol{y}_{\text{obs}} \mid \boldsymbol{\xi})p(\boldsymbol{\xi}). \tag{10}$$

Algorithm 1 Hamiltonian Monte Carlo (HMC)

Require: initial states $\boldsymbol{\xi}^0$, time step size δt, leapfrog steps I, total running steps J
  for $j = 0, 1, \ldots, (J-1)$ **do**
    Sample r^j from $\mathcal{N}(0, \boldsymbol{M})$
    $\boldsymbol{\xi}_0 \leftarrow \boldsymbol{\xi}^j$
    $r_0 \leftarrow r^j$
    for $i = 0, 1, \ldots, (I-1)$ **do**
      $\boldsymbol{r}_i \leftarrow \boldsymbol{r}_i - \frac{\delta t}{2} \nabla V(\boldsymbol{\xi}_i)$
      $\boldsymbol{\xi}_{i+1} \leftarrow \boldsymbol{\xi}_i + \delta t M^{-1} \boldsymbol{r}_i$
      $\boldsymbol{r}_{i+1} \leftarrow \boldsymbol{r}_i - \frac{\delta t}{2} \nabla V(\boldsymbol{\xi}_{i+1})$
    end for
    Sample $\boldsymbol{p}$ from $\mathcal{U}(0, 1)$
    $\alpha \leftarrow \min\left[1, \exp\left(H(\boldsymbol{\xi}_I, \boldsymbol{r}_I) - H\left(\boldsymbol{\xi}^j, \boldsymbol{r}^j\right)\right)\right]$
    if $p \geq \alpha$ **then**
      $\boldsymbol{\xi}^{j+1} \leftarrow \boldsymbol{\xi}_I$
    else
      $\boldsymbol{\xi}^{j+1} \leftarrow \boldsymbol{\xi}^j$
    end if
  end for
  Return: $\boldsymbol{\xi}^1, \ldots, \boldsymbol{\xi}^J$

Assuming the prior distribution of the parameters as $\boldsymbol{\xi} \sim \mathcal{N}(\xi_0, C_0)$, the likelihood can be expressed as

$$p(\boldsymbol{y}_{\text{obs}} \mid \boldsymbol{\xi}) \propto \exp\left(-\frac{\left\| R^{-1/2}(\boldsymbol{y} - \mathcal{G}(\boldsymbol{\xi})) \right\|_2^2}{2}\right). \tag{11}$$

To facilitate iterative inversion, we reformulate the Bayesian inverse problem as an artificial dynamical system:

$$\begin{aligned} \boldsymbol{\xi}_i &= \boldsymbol{\xi}_{i-1} + \boldsymbol{\epsilon}_i, \quad \boldsymbol{\epsilon}_i \sim \mathcal{N}(0, Q), \\ \boldsymbol{y}_i &= \mathcal{G}(\boldsymbol{\xi}_i) + \boldsymbol{\eta}_i, \quad \boldsymbol{\eta}_i \sim \mathcal{N}(0, R). \end{aligned} \tag{12}$$

where $\boldsymbol{\epsilon}_i$ represents an artificial parameter noise with covariance Q and $\boldsymbol{\eta}_i$ denotes the observation error with covariance R. In this formulation, the parameters are treated as state variables that evolve incrementally through the iterative process, while the observation equations remain unchanged. To efficiently approximate the posterior distribution, EKI employs the Kalman gain formula for Gaussian posterior distributions to iteratively update the sample set as in Ref. [9]. Let $\left\{\boldsymbol{\xi}_0^j\right\}_{j=1}^J$ denote the initial ensemble of J members drawn from the prior distribution. Then, at iteration i, the j-th ensemble member is updated as

$$\boldsymbol{\xi}_{i+1}^j = \boldsymbol{\xi}_i^j + \boldsymbol{\epsilon}_i^j + C_i^{\xi y}\left(C_i^{yy} + R\right)^{-1}\left(\boldsymbol{y}_{\text{obs}} + \boldsymbol{\eta}_i^j - \mathcal{G}\left(\boldsymbol{\xi}_i^j\right)\right), \tag{13}$$

where $\boldsymbol{\epsilon}_i^j$ and $\boldsymbol{\eta}_i^j$ represent the corresponding parameter and the observation noise. The sample covariances are defined as

$$\begin{aligned} C_i^{yy} &= \frac{1}{J-1}\sum_{j=1}^{J}\left(\boldsymbol{y}_i^j - \overline{\boldsymbol{y}}_i\right)\left(\boldsymbol{y}_i^j - \overline{\boldsymbol{y}}_i\right)^T, \\ C_i^{\xi y} &= \frac{1}{J-1}\sum_{j=1}^{J}\left(\boldsymbol{\xi}_i^j - \overline{\boldsymbol{\xi}}_i\right)\left(\boldsymbol{y}_i^j - \overline{\boldsymbol{y}}_i\right)^T. \end{aligned} \tag{14}$$

At each iteration, the ensemble is updated according to the EKI scheme, producing a new set of parameter samples that progressively approximate the target posterior distribution.

2.4. Quantum Model

The quantum model $f_{\boldsymbol{\theta}}(\boldsymbol{x})$ is formally defined as the expectation value of an observable O with respect to a state evolved by the quantum circuit [14] $U(\boldsymbol{x}, \boldsymbol{\theta})$ as follows:

$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \langle 0|U^{\dagger}(\boldsymbol{x}, \boldsymbol{\theta}) O U(\boldsymbol{x}, \boldsymbol{\theta})|0\rangle, \tag{15}$$

where $|0\rangle$ denotes the initial quantum state and $U(\boldsymbol{x}, \boldsymbol{\theta})$ is the quantum circuit, which depends on the input $\boldsymbol{x}$ and the parameter $\boldsymbol{\theta}$, and takes the following form:

$$U(\boldsymbol{x}, \boldsymbol{\theta}) = W^{(L+1)}(\boldsymbol{\theta}) S(\boldsymbol{x}) W^{(L)}(\boldsymbol{\theta}) \ldots W^{(2)}(\boldsymbol{\theta}) S(\boldsymbol{x}) W^{(1)}(\boldsymbol{\theta}). \tag{16}$$

The quantum circuit is composed of sequential layers L, where each layer includes a data-encoding circuit block $S(\boldsymbol{x})$ and a trainable variational circuit block $W(\boldsymbol{\theta})$. An example quantum circuit with 3 qubits and 2 sequential layers is shown in Figure 1.

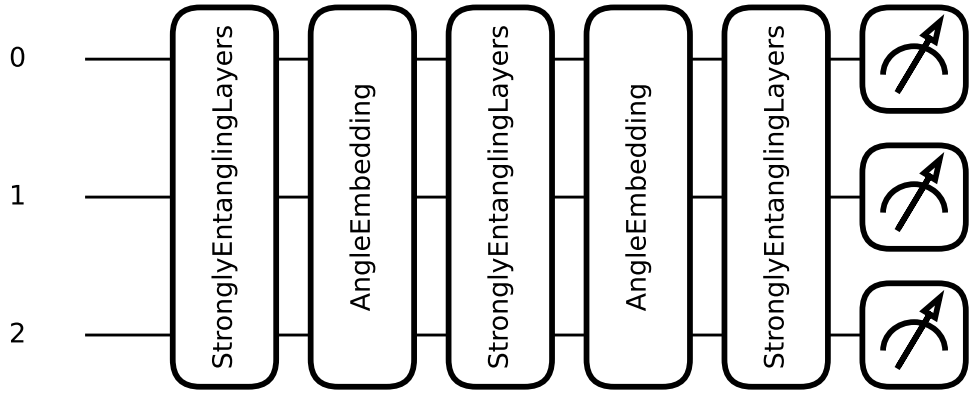


Figure 1. Quantum model architecture with 3 qubits, 2 sequential layers, and a measurement layer.

The trainable and encoding layers of the quantum circuit are constructed using rotational gates [23]. The matrix representation of these gates is given below:

$$\begin{aligned} R_X(\theta) &= \begin{pmatrix} \cos\left(\frac{\theta}{2}\right) & -i\sin\left(\frac{\theta}{2}\right) \\ -i\sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) \end{pmatrix}, \\ R_Y(\theta) &= \begin{pmatrix} \cos\left(\frac{\theta}{2}\right) & -\sin\left(\frac{\theta}{2}\right) \\ \sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) \end{pmatrix}, \\ R_Z(\theta) &= \begin{pmatrix} \exp\left(-i\frac{\theta}{2}\right) & 0 \\ 0 & \exp\left(i\frac{\theta}{2}\right) \end{pmatrix}. \end{aligned} \tag{17}$$

Applying these rotation gates to the basis states $|0\rangle$ and $|1\rangle$ produces:

$$
\begin{aligned}
R_X(\theta)|0\rangle &= \cos\left(\frac{\theta}{2}\right)|0\rangle - i\sin\left(\frac{\theta}{2}\right)|1\rangle,\\
R_X(\theta)|1\rangle &= -i\sin\left(\frac{\theta}{2}\right)|0\rangle + \cos\left(\frac{\theta}{2}\right)|1\rangle,\\
R_Y(\theta)|0\rangle &= \cos\left(\frac{\theta}{2}\right)|0\rangle + \sin\left(\frac{\theta}{2}\right)|1\rangle,\\
R_Y(\theta)|1\rangle &= -\sin\left(\frac{\theta}{2}\right)|0\rangle + \cos\left(\frac{\theta}{2}\right)|1\rangle,\\
R_Z(\theta)|0\rangle &= \left[\cos\left(\frac{\theta}{2}\right) - i\sin\left(\frac{\theta}{2}\right)\right]|0\rangle,\\
R_Z(\theta)|1\rangle &= \left[\cos\left(\frac{\theta}{2}\right) + i\sin\left(\frac{\theta}{2}\right)\right]|1\rangle.
\end{aligned}
\tag{18}
$$

The data-encoding block $S(\boldsymbol{x})$ is mathematically defined as a tensor product of rotations R_z:

$$S(\boldsymbol{x}) = \bigotimes_{i=1}^{n} R_Z(x_i) \tag{19}$$

where n is the number of qubits and x_i is the i-th component of the input vector $\boldsymbol{x}$. Applying the encoding block to the initial n-qubit state $\bigotimes_{i=1}^{n}|0\rangle$, we obtain the encoded state $|\boldsymbol{x}\rangle = \bigotimes_{i=1}^{n}|x_i\rangle$ as follows:

$$|\boldsymbol{x}\rangle = S(\boldsymbol{x})\left(\bigotimes_{i=1}^{n}|0\rangle\right) = \bigotimes_{i=1}^{n}(R_Z(x_i)|0\rangle) = \bigotimes_{i=1}^{n}\left(\left[\cos\left(\frac{x_i}{2}\right) - i\sin\left(\frac{x_i}{2}\right)\right]|0\rangle\right). \tag{20}$$

We use strongly entangling layers [24] as the trainable circuit block and training blocks $W(\boldsymbol{\theta})$ depend on the parameters $\boldsymbol{\theta}$ that can be classically optimized. A strongly entangling layer with three qubits that contains 9 trainable variables takes the form as in Figure 2.

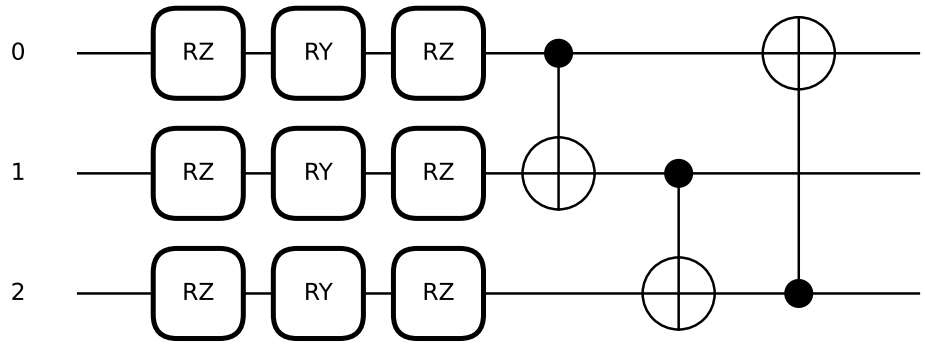


Figure 2. Circuit layout of a three-qubit strongly entangling layer using rotation gates and CNOT gates.

Applying the strongly entangling layer to the encoded state produces the following result:

$$W(\boldsymbol{\theta})|\boldsymbol{x}\rangle = U_{CNOT}^{31}U_{CNOT}^{23}U_{CNOT}^{12}\left(\bigotimes_{i=1}^{3} R_Z\left(\theta_i^1\right)R_Y\left(\theta_i^2\right)R_Z\left(\theta_i^3\right)|x_i\rangle\right), \tag{21}$$

where U_{CNOT}^{ij} denotes a controlled-NOT gate acting on qubit i (control) and j (target). Applying the quantum circuit $U(\boldsymbol{x},\boldsymbol{\theta})$ to the initial state $|0\rangle$ produces the final state $|\hat{\boldsymbol{x}}\rangle$:

$$|\hat{\boldsymbol{x}}\rangle = U(\boldsymbol{x},\boldsymbol{\theta})|0\rangle. \tag{22}$$

This final state encodes both the classical input $\boldsymbol{x}$ and the variational parameters $\boldsymbol{\theta}$, and serves as the basis for subsequent measurements to obtain the model output. Then the output of the quantum model is obtained by measuring a Hermitian observable O, whose expectation value defines the model prediction [25]:

$$\langle O \rangle = \langle \hat{\boldsymbol{x}} | O | \hat{\boldsymbol{x}} \rangle, \tag{23}$$

where O is a Hermitian operator representing the measured observable. We choose O to be a tensor product of Pauli-Z operators $\bigotimes_{i=1}^{n} Z_i$, which allows efficient readout of the model output from the quantum state. In practice, $\langle O \rangle$ is estimated by repeated measurements of the observable O on multiple runs of the circuit.

2.5. Quantum-Encodable Bayesian PINNs Trained via Classical Ensemble Kalman Inversion

Gradient-based optimization of QNNs often encounters the barren plateau phenomenon, where the variance of gradients decays exponentially with the number of qubits or the depth of the circuit. As a result, randomly initialized circuits often produce gradients that are effectively zero, causing extremely slow or even stalled learning, particularly in deep or noisy quantum circuits. As a gradient-free update method, EKI naturally circumvents these difficulties by reducing the need for explicit gradient computation, thus avoiding the gradient concentration problems associated with barren plateaus, which enhances the training stability and accelerates the overall optimization of the QNN.

Motivated by these advantages, we incorporate QNNs into the EKI–BPINNs framework as surrogate models to solve PDE inverse problems. In this framework, the QNN surrogate approximates the forward solution of the governing PDE. Its parameters and the unknown PDE parameters are treated as part of the ensemble, which evolves during the inversion process. The EKI update infers the unknown physical parameters by assimilating noisy observational data while maintaining ensemble diversity. Furthermore, to fully exploit the input data for the QNN, we first perform a classical preprocessing step that constructs the network inputs as learnable linear combinations of the raw PDE data. Specifically, the QNN input H_{in} is defined as $H_{in} = H_0 W + b$, where H_0 represents the raw data, including residual data, boundary-condition data and solution measurement, and W and b are learnable parameters, which serve as a classical linear layer to map the raw data into a vector whose dimension matches the number of qubits used in the quantum circuit.

Algorithm 2 summarizes the complete training and inversion procedure. In this study, all computational stages are implemented and executed on classical hardware, where the QNN is simulated using the PennyLane v0.43 [26] quantum circuit simulator. In the current framework, the inversion algorithm, ensemble perturbation, and residual-based updates are entirely classical, while the quantum role is confined to the QNN-based surrogate representation and its circuit evaluation.

The classical–quantum interaction is organized as an iterative parameter-update scheme. In each step, the BPINNs input is mapped to a quantum state using angle encoding, and the parameterized variational layers transform the state according to the current QNN weights. The circuit is evaluated repeatedly with several measurement shots to estimate expectation values, which serve as the circuit output. These outputs are passed to the classical EKI routine, where the residual between the observations and the quantum surrogate predictions is used to generate the next ensemble update.

Within B-PINNs, the parameters $\boldsymbol{\xi} = \{\boldsymbol{\theta}, \boldsymbol{\lambda}\}$ are considered, with the forward operator $\mathcal{G}$ defining the model mapping. The quantity $\boldsymbol{y}_i^j$ represents the result of evaluating the forward model $\mathcal{G}(\boldsymbol{\xi}_i^j)$ in one step, which is defined as

$$\boldsymbol{y}_i^j = \mathcal{G}(\boldsymbol{\xi}_i^j) = \begin{bmatrix} u(\boldsymbol{x};\boldsymbol{\theta}_i^j) \\ \mathcal{N}_{\boldsymbol{x}}(u(\boldsymbol{x};\boldsymbol{\theta}_i^j);\boldsymbol{\lambda}_i^j) \\ \mathcal{B}_{\boldsymbol{x}}(u(\boldsymbol{x};\boldsymbol{\theta}_i^j);\boldsymbol{\lambda}_i^j) \end{bmatrix}. \tag{24}$$

Algorithm 2 Quantum-Encodable Bayesian PINNs trained via Classical Ensemble Kalman Inversion (QEKI)

Require: Observations y_{obs}, initial J ensemble states $\left\{\boldsymbol{\xi}_0^j\right\}_{j=1}^J$, observation covariance R, parameter covariance Q, training data points $\boldsymbol{x}$, iteration index $i = 0$.
while not converge **do**
  for $j = 1, 2, \ldots, J$ **do**
    Sample $\boldsymbol{\epsilon}_i^j$ from $\mathcal{N}(0, Q)$
    Update each ensemble state $\boldsymbol{\xi}_i^j \leftarrow \boldsymbol{\xi}_i^j + \boldsymbol{\epsilon}_i^j$
    $(\boldsymbol{\theta}_i^j, \boldsymbol{\lambda}_i^j) \leftarrow \boldsymbol{\xi}_i^j$
    Apply quantum circuit $|\hat{\boldsymbol{x}}\rangle \leftarrow U(\boldsymbol{x}, \boldsymbol{\theta})|0\rangle$
    Evaluate the expectation value $\langle O\rangle_i^j \leftarrow \langle \hat{\boldsymbol{x}}|O|\hat{\boldsymbol{x}}\rangle$
    $\boldsymbol{y}_i^j \leftarrow \mathcal{N}_{\boldsymbol{x}}(\langle O\rangle_i^j; \boldsymbol{\lambda}_i^j)$
  end for
  $C_i^{yy} \leftarrow \frac{1}{J-1}\sum_{j=1}^J \left(\boldsymbol{y}_i^j - \overline{\boldsymbol{y}}_i\right)\left(\boldsymbol{y}_i^j - \overline{\boldsymbol{y}}_i\right)^T$
  $C_i^{\xi y} \leftarrow \frac{1}{J-1}\sum_{j=1}^J \left(\boldsymbol{\xi}_i^j - \overline{\boldsymbol{\xi}}_i\right)\left(\boldsymbol{y}_i^j - \overline{\boldsymbol{y}}_i\right)^T$
  for $j = 1, 2, \ldots, J$ **do**
    Sample $\boldsymbol{\eta}_i^j$ from $\mathcal{N}(0, R)$
    $\boldsymbol{\xi}_{i+1}^j \leftarrow \boldsymbol{\xi}_i^j + C_i^{\xi y}\left(C_i^{yy} + R\right)^{-1}\left(\boldsymbol{y}_{\text{obs}} + \boldsymbol{\eta}_i^j - \boldsymbol{y}_i^j\right)$
  end for
end while
$\hat{I} \leftarrow i + 1$
Return: $\boldsymbol{\xi}_{\hat{I}}^1, \ldots, \boldsymbol{\xi}_{\hat{I}}^J$

Correspondingly, the observations defined in (2) can be formulated as follows:

$$\boldsymbol{y}_{\text{obs}} = \begin{bmatrix} \{u^i\}_{i=1}^{N_u} & \{f^i\}_{i=1}^{N_f} & \{b^i\}_{i=1}^{N_b} \end{bmatrix}^T. \tag{25}$$

The physics-based residuals are evaluated via PennyLane's interface, which maps the quantum circuit to a classically differentiable node. This setup enables the use of classical automatic differentiation while maintaining compatibility with native quantum gradient methods, such as the parameter-shift rule, for future execution on quantum processing units.

The choice of covariance matrices Q and R can significantly affect the performance of the algorithm. Previous studies have explored automatic estimation techniques for these matrices. In this work, following the strategy proposed in Ref. [6], we set R to be the covariance matrix as defined in (5):

$$R = \begin{bmatrix} \sigma_u^2 I_{N_u} & 0 & 0 \\ 0 & \sigma_f^2 I_{N_f} & 0 \\ 0 & 0 & \sigma_b^2 I_{N_b} \end{bmatrix} \tag{26}$$

An appropriate choice of Q is essential to preserve ensemble spread, so we set Q to be

$$Q = \begin{bmatrix} \sigma_\theta^2 I_{N_\theta} & 0 \\ 0 & \sigma_\lambda^2 I_{N_\lambda} \end{bmatrix} \tag{27}$$

where σ_θ and σ_λ represent the standard deviations of the artificial process noise introduced to preserve ensemble spread for the QNN parameters $\boldsymbol{\theta}$ and the physical parameters $\boldsymbol{\lambda}$, respectively.

To determine when QEKI iterations should terminate, we employ a discrepancy-based stopping rule inspired by classical iterative regularization methods. The basic idea is to monitor how well the prediction of the current ensemble-averaged model matches the observed data, measured by a weighted residual norm as defined in Ref. [6]. Let

$$D_i = \left\| R^{-1/2} \left(\boldsymbol{y}_{\text{obs}} - \frac{1}{J} \sum_{j=1}^{J} \boldsymbol{y}_i^j \right) \right\|, \tag{28}$$

denote the discrepancy at iteration i, where $\boldsymbol{y}_i^j$ is the predicted observation produced by the j-th ensemble member at iteration i. Specifically, we define a sliding window of length W and terminate the iteration once the discrepancy no longer exhibits sufficient improvement, i.e.,

$$\max_{j \in \{i-W,\ldots,i\}} \frac{|D_j - D_i|}{D_i} < \tau. \tag{29}$$

Complexity Analysis. The computational cost of QEKI is lower than that of HMC, as it performs direct ensemble updates. The per-iteration computational complexity of QEKI is given by:

$$O\left(N_y^3 + JN_yN_\xi + JN_y^2\right). \tag{30}$$

This complexity can be decomposed into the following components:

(1) $O(JN_y^2)$—Cost of constructing the observation covariance matrix C_i^{yy}
(2) $O(JN_yN_\xi)$—Cost of constructing the observation covariance matrix $C_i^{\xi y}$
(3) $O(N_y^3 + JN_yN_\xi + JN_y^2)$—Cost of updating all ensemble members via the Kalman gain, $C_i^{\xi y}(C_i^{yy} + R)^{-1}(\boldsymbol{y}_{\text{obs}} + \boldsymbol{\eta}_i^j - \boldsymbol{y}_i^j)$.

In conclusion, the computational analysis indicates that the primary bottleneck of QEKI lies in the observation dimension N_y, specifically due to the $O(N_y^3)$ cost associated with the matrix inversion in the Kalman gain. Conversely, the algorithm exhibits linear scalability with respect to both the high-dimensional parameter space N_ξ and the ensemble size J. This structural characteristic allows for rapid ensemble updates without prohibitive computational costs.

This completes the description of the proposed QEKI framework. The next section presents numerical experiments that demonstrate its performance on representative PDE inverse problems.

3. Experiments

In this section, we present the experimental setup and numerical studies to evaluate the performance of the proposed QEKI framework. To ensure a fair and systematic comparison, we use the classical HMC-based B-PINNs as a baseline. This allows us to evaluate how well the quantum-encodable architecture can reproduce the posterior distribution under a well-understood classical inversion procedure. We note that due to current quantum-training bottlenecks and hardware limitations, QEKI does not achieve the same training speed as classical EKI-trained DNN surrogates. Nevertheless, a key advantage of the proposed approach

is that it can maintain comparable posterior accuracy while using fewer trainable parameters, demonstrating the representational efficiency of the quantum-encodable architecture. All computations are executed on the open-source PennyLane v0.43 [26] and JAX 0.8.0 [27] platforms on a single GPU (NVIDIA GeForce RTX 4090 with 24 GB of memory).

In the HMC case, we adopt the variant of HMC described in [28]. The surrogate model is a neural network with two hidden layers, each containing 50 neurons and equipped with the activation function tanh. The number of neural network parameters is listed for each experiment. For the HMC implementation, the leapfrog integration step is set to $I = 50$, the initial time step is $\delta t = 0.1$, the burn-in period consists of 1000 steps, and a total of 2000 posterior samples are collected.

In the EKI case, we employ a QNN architecture, as described in Ref. [14], consisting of four qubits and six entangling layers, while in Experiment 3, we use a circuit with six qubits and eight entangling layers to accommodate the increased model complexity introduced by the nonlinear advection term in the Burgers equation. The size of the ensemble is set to $J = 200$, and the initial ensemble states $\left\{\boldsymbol{\zeta}_0^j\right\}_{j=1}^J$ are drawn from the prior distribution, the standard deviations of artificial dynamics of the parameters are chosen to be $\sigma_\lambda = 0.01$ and $\sigma_\theta = 0.01$. For the stopping rule, we choose the size of the sliding window to be $W = 25$ and the tolerance $\tau = 0.05$.

Synthetic observation data are generated by first solving the target PDEs and then adding Gaussian noise with two different levels ($\sigma_u = \sigma_f = \sigma_b$ = 0.1 and 0.01) to test robustness against observation uncertainty. For the 1D problems, both the solution measurements and the PDE residual points are placed on uniformly spaced grids. For the 2D examples, the solution measurements are randomly sampled in the physical domain, while the residual points inside the domain are generated using Latin hypercube sampling, and the boundary points are uniformly sampled along the boundary. The priors are all standard Gaussian distributions with zero mean.

To assess the accuracy of the solution and the estimation of the physical parameter, we compute the following error:

$$e_{\boldsymbol{u}} = \frac{\|\boldsymbol{u} - \bar{\boldsymbol{u}}\|_{L_2}}{\|\boldsymbol{u}\|_{L_2}}, \quad e_\lambda = \frac{\|\lambda - \bar{\lambda}\|_{L_1}}{\|\lambda\|_{L_1}}, \tag{31}$$

where $\boldsymbol{u}$ and λ are the reference solution and reference physical parameter, $\bar{\boldsymbol{u}}$ and $\bar{\lambda}$ are the sample means of the approximate solution and approximate physical parameter. In all experiments considered in this work, we restrict λ to a scalar parameter. To evaluate computational performance, we compare the mean walltime of QEKI and HMC over 10 independent trials. We further compare the number of trainable parameters, with the numbers in parentheses indicating the total number of parameters in the linear classical layers at the input of the QNN.

3.1. One-Dimensional Nonlinear Poisson Equation

We first consider the 1D nonlinear Poisson problem:

$$\lambda u_{xx} + k \tanh(u) = f, \quad x \in [-0.7, 0.7]. \tag{32}$$

In this experiment, we set $\lambda = 0.01$, and set $k = 0.7$ as the unknown physical parameter. The forward problem admits the analytical solution $u(x) = sin^3(x)$, from which the source term and the boundary conditions are exactly derived. The inverse task is to recover the parameter k and the solution $u(x)$, together with their associated uncertainty estimates. To enforce the PDE constraints, we employ 8 interior points, 2 boundary points, and 32 collocation points.

As shown in Table 1, both QEKI and HMC achieve an accurate recovery of the physical parameter k, with the true value $k = 0.7$ consistently lying within one standard deviation band of their posterior mean estimates in both noise settings. This indicates that both Bayesian frameworks can provide statistically reliable uncertainty quantification for the inverse problem.

Table 1. Section 3.1: Sample mean and standard deviation of parameter k for QEKI and HMC for $\sigma_u = 0.01, 0.1$ noise levels. The true value of k is 0.7.

		k (Mean $\pm$ Std)
$\sigma_u = 0.01$	QEKI	0.699 ± 0.006
	HMC	0.702 ± 0.007
$\sigma_u = 0.1$	QEKI	0.689 ± 0.011
	HMC	0.687 ± 0.018

Figure 3 further compares the sample mean and standard deviation of the reconstructed surrogate solutions; the QEKI results show a consistently smaller posterior variance.

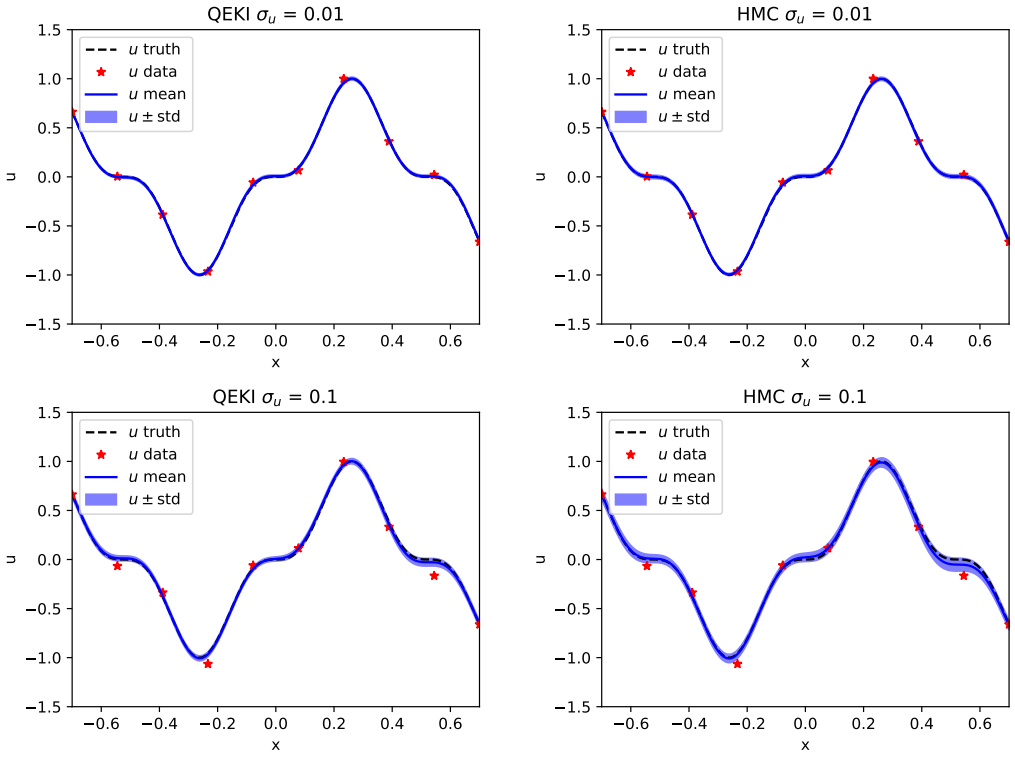


Figure 3. Section 3.1: Reference solution, observation samples, sample mean and standard deviation by QEKI and HMC for $\sigma_u = 0.01, 0.1$ noise levels.

Table 2 further reports the relative errors in the estimated means of u and k, together with the computational walltime and the number of trainable parameters for each method. Notably, QEKI achieves comparable or better accuracy while requiring substantially fewer trainable parameters and reduced computational cost, highlighting its potential as a lightweight yet effective quantum-assisted inversion framework.

Table 2. Section 3.1: Relative errors e_u of the forward solution u and e_k of parameter k for the noise levels σ_u = 0.01, 0.1, average walltime and trainable parameters.

		e_u	e_k	Walltime (s)	Trainable Parameters
$\sigma_u = 0.01$	QEKI	1.23%	0.14%	13.01	85 (+8)
	HMC	1.13%	0.15%	45.32	5252
$\sigma_u = 0.1$	QEKI	8.03%	1.57%	13.11	85 (+8)
	HMC	9.21%	1.85%	46.02	5252

3.2. Two-Dimensional Nonlinear Diffusion–Reaction Equation

Here, we consider the following 2D nonlinear PDE:

$$\begin{aligned} \lambda \Delta u + k u^2 &= f, \quad (x, y) \in [-1, 1]^2, \\ u(x, -1) &= u(x, 1) = 0, \\ u(-1, y) &= u(1, y) = 0. \end{aligned} \tag{33}$$

In this experiment, we set $\lambda = 0.01$ and set $k = 1$ as the unknown physical parameter to be inferred. The forward problem admits the analytical solution $u(x) = sin(\pi x)sin(\pi y)$, from which the source term and the boundary conditions are analytically derived. The inverse problem aims to recover both the parameter k and the solution $u(x)$ with uncertainty estimates. To enforce the PDE constraints, we employ 100 interior points, 100 boundary points, and 100 collocation points. The true solution and the observation points are illustrated in Figure 4.

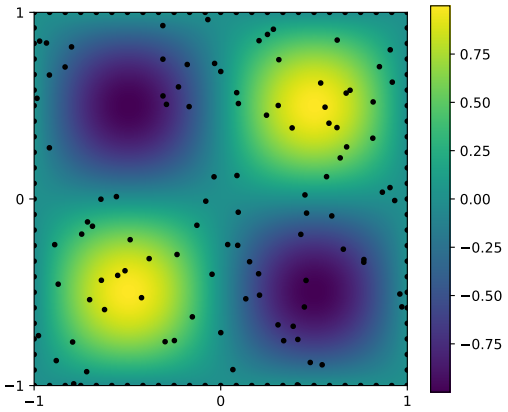


Figure 4. Section 3.2: Observation points of the forward solution and the corresponding boundary measurements (represented by black dots).

Table 3 shows that both QEKI and HMC successfully recover the physical parameter k, with the true value $k = 1$ consistently lying within the two standard deviation bands of their posterior mean estimates in both noise settings. This indicates that both Bayesian frameworks provide reliable uncertainty quantification for the inverse problem.

Table 3. Section 3.2: Sample mean and standard deviation of the parameter k for QEKI and HMC for σ_u = 0.01, 0.1 noise levels. The true value of k is 1.

		k (Mean $\pm$ Std)
$\sigma_u = 0.01$	QEKI	0.992 ± 0.008
	HMC	0.988 ± 0.012
$\sigma_u = 0.1$	QEKI	1.029 ± 0.022
	HMC	0.964 ± 0.035

Figure 5 compares the sample mean and standard deviation of the surrogate solutions obtained by the two approaches. The QEKI mean closely matches the reference solution, demonstrating its effectiveness in reconstructing the forward solution, while HMC similarly captures the overall trend but exhibits slightly larger posterior variance.

Table 4 shows the relative error of the estimated means for both u and k, along with the computational walltime and the number of trainable parameters for each method. Notably, QEKI achieves comparable or slightly better accuracy while requiring fewer trainable parameters and reduced computational cost, highlighting its potential as a computationally efficient quantum-assisted inversion framework. These results collectively demonstrate that QEKI can provide accurate and stable posterior estimates for both the physical parameter and the surrogate solution, even under different noise levels.

Table 4. Section 3.2: Relative errors e_u of the forward solution u and e_k of the unknown parameter k for noise levels σ_u = 0.01, 0.1, including average walltime and number of trainable parameters.

		e_u	e_k	Walltime (s)	Trainable Parameters
$\sigma_u = 0.01$	QEKI	1.03%	0.88%	23.42	85 (+12)
	HMC	1.21%	1.21%	51.22	5302
$\sigma_u = 0.1$	QEKI	2.77%	3.41%	24.52	85 (+12)
	HMC	2.82%	3.62%	52.03	5302

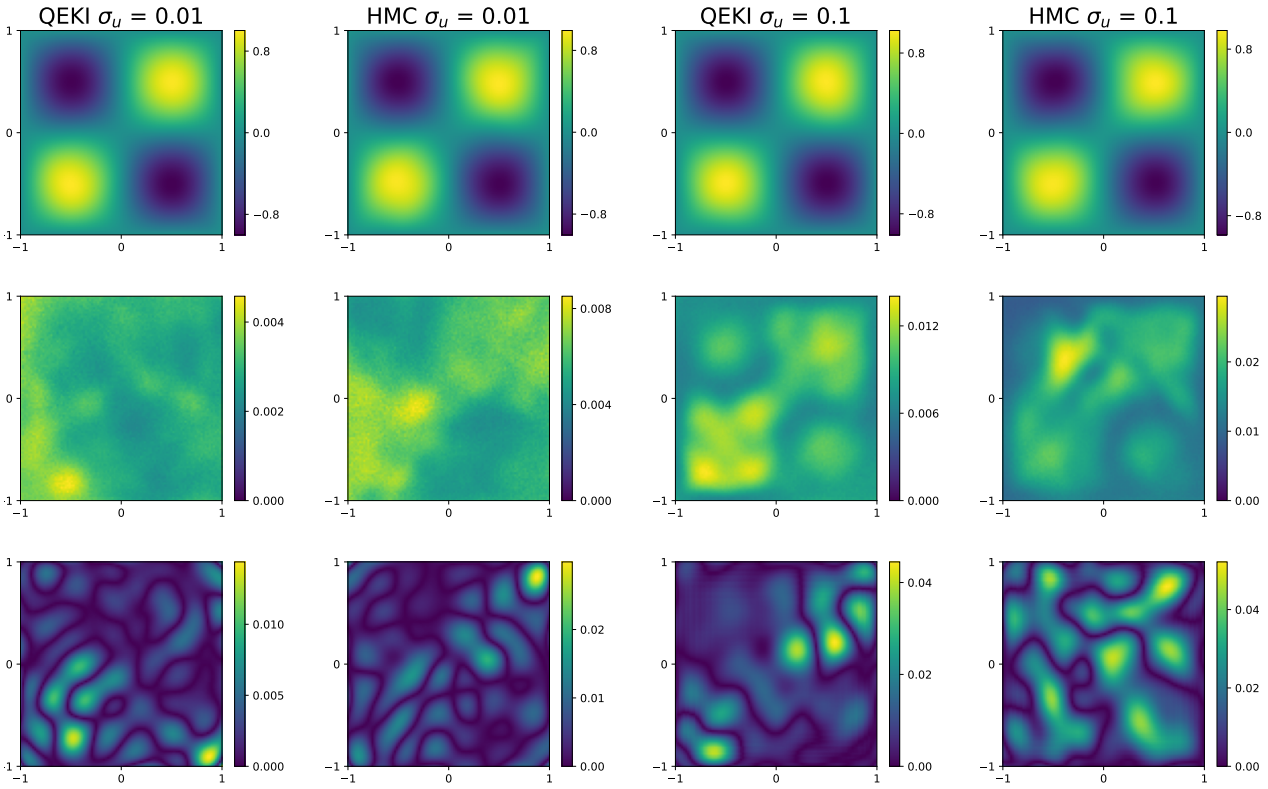


Figure 5. Section 3.2: Comparison of QEKI and HMC under different noise levels $\sigma_u = 0.01, 0.1$. The prediction values obtained by both methods (**top row**), the corresponding standard deviations (**middle row**), the absolute error between the ground-truth solution (**bottom row**).

3.3. Burgers Equation

Here, we consider the following Burgers equation as presented in Ref. [15]:

$$u_t + uu_x = \nu u_{xx}, \tag{34}$$

with a viscosity of $\nu = 0.01$,which we set to be the unknown physical parameter, in a computational domain of $(x,t) = [-1,1] \times [0,1]$. The exact solution under the Dirichlet boundary condition is given by

$$u(x,t) = \frac{\frac{x}{t+1}}{1+\sqrt{\frac{t+1}{t_0}}e^{\frac{x^2}{4\nu(t+1)}}}, \tag{35}$$

with $t_0 = \exp(\frac{1}{8\mu})$. The initial and boundary conditions can be derived exactly. To enforce the PDE constraints, we employ 50 initial points, 100 boundary points, 200 collocation points, and 200 interior points. The true solution and the observation points are illustrated in Figure 6.

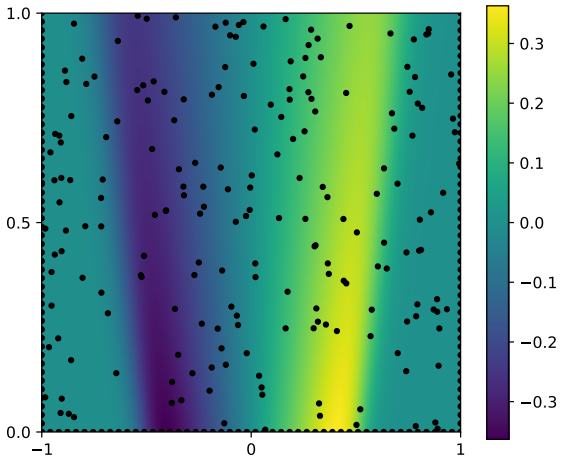


Figure 6. Section 3.3: Observation points of the forward solution and the corresponding initial and boundary measurements (represented by black dots).

Table 5 shows that both QEKI and HMC successfully recover the physical parameter ν, with the true value $\nu = 0.01$ consistently lying within the one standard deviation interval of the posterior mean under both noise levels. This indicates that both Bayesian inversion frameworks remain effective even in the presence of nonlinearity introduced by the Burgers equation.

Table 5. Section 3.3: Sample mean and standard deviation of the parameter ν for QEKI and HMC for σ_u = 0.01, 0.1 noise levels. The true value of ν is 0.01.

		ν $(\times 10^3)$ **(Mean $\pm$ Std)**
$\sigma_u = 0.01$	QEKI	10.257 ± 0.327
	HMC	9.719 ± 0.502
$\sigma_u = 0.1$	QEKI	10.614 ± 0.823
	HMC	10.740 ± 0.908

Figure 7 further compares the surrogate predictions obtained from the two methods. The QEKI produces a mean solution that closely matches the reference, while the HMC captures the overall trend.

In addition, Table 6 summarizes the relative errors of the posterior mean estimates for both u and ν, together with the computational wall-time and the number of trainable parameters for each method. Notably, although the QNN architecture is enlarged for this nonlinear problem, the computational speed of QEKI experiences a slight decrease due to quantum hardware limitations, yet it still requires fewer trainable parameters compared to

HMC. These results indicate that QEKI can reliably provide accurate posterior estimates for both the physical parameter and the surrogate solution, even in more challenging nonlinear scenarios, while maintaining overall computational efficiency.

Table 6. Section 3.3: Relative errors e_u of the forward solution u and e_v of the unknown parameter v for noise levels $\sigma_u = 0.01, 0.1$, including average walltime and number of trainable parameters.

		e_u	e_v	Walltime (s)	Trainable Parameters
$\sigma_u = 0.01$	QEKI	1.41%	2.57%	44.88	163 (+18)
	HMC	1.58%	2.81%	56.33	5302
$\sigma_u = 0.1$	QEKI	4.22%	6.41%	45.03	163 (+18)
	HMC	4.72%	7.41%	56.09	5302

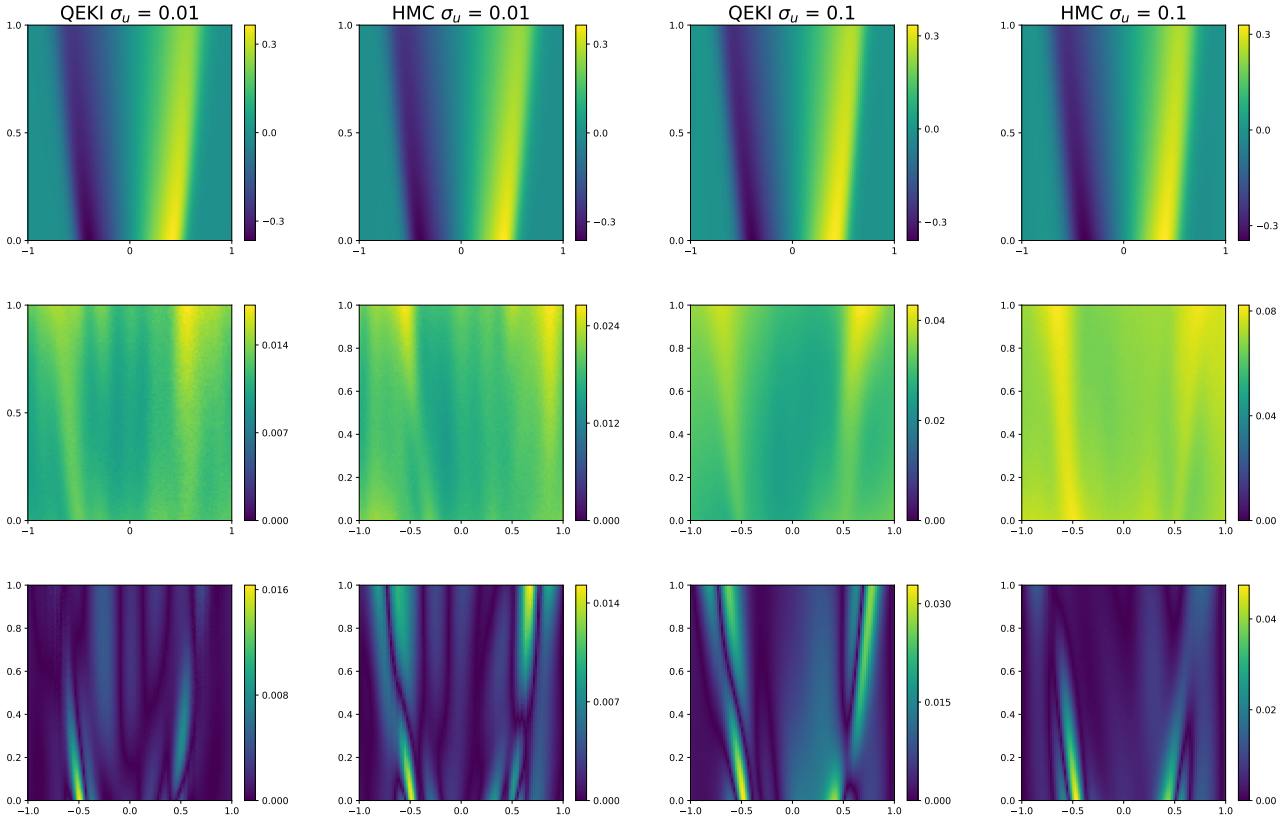


Figure 7. Section 3.3: Comparison of QEKI and HMC under different noise levels $\sigma_u = 0.01, 0.1$. The prediction values obtained by both methods (**top row**), the corresponding standard deviations (**middle row**), the absolute error between the ground-truth solution (**bottom row**).

3.4. Capacity Analysis

The focus of this experiment is to assess representational efficiency under a similar number of trainable parameters for the classical DNN-based method in comparison with the proposed QEKI architecture. In particular, we first examine whether the DNN model remains trainable with which architectures are constrained to a similar parameter budget as QEKI. In practice, we observe that the DNN-based method exhibits insufficient expressive power under this equal-capacity constraint and fails to achieve stable convergence, resulting in unreliable posterior estimates. In contrast, the QEKI model remains trainable and produces stable posterior inference under the same parameter budget. For this reason, additional DNN-based models with increased parameter counts are included to evaluate

how much model capacity is required for classical architectures to recover comparable posterior accuracy and training behavior.

Specifically, we consider neural network architectures of three different sizes, corresponding to two-hidden-layer fully connected networks with 10, 30, and 50 neurons per hidden layer, resulting in 152, 1052, and 2752 trainable parameters, respectively. For each network configuration, both EKI and HMC are employed as inference engines. All experiments are conducted on the Burgers equation, as in Section 3.3, with the observation noise level fixed to 0.01. In all cases, the associated hyperparameters are selected following standard practice and kept consistent across models to ensure a fair comparison.

As shown in Table 7, we observe that networks with 10 and 30 neurons per layer fail to achieve stable convergence, while the network with 50 neurons per layer successfully converges and produces posterior estimates comparable to those obtained with QEKI, as shown in Figure 8. These results indicate that the QEKI architecture can maintain meaningful posterior inference under a much more restrictive parameter budget, highlighting the representational efficiency of QEKI.

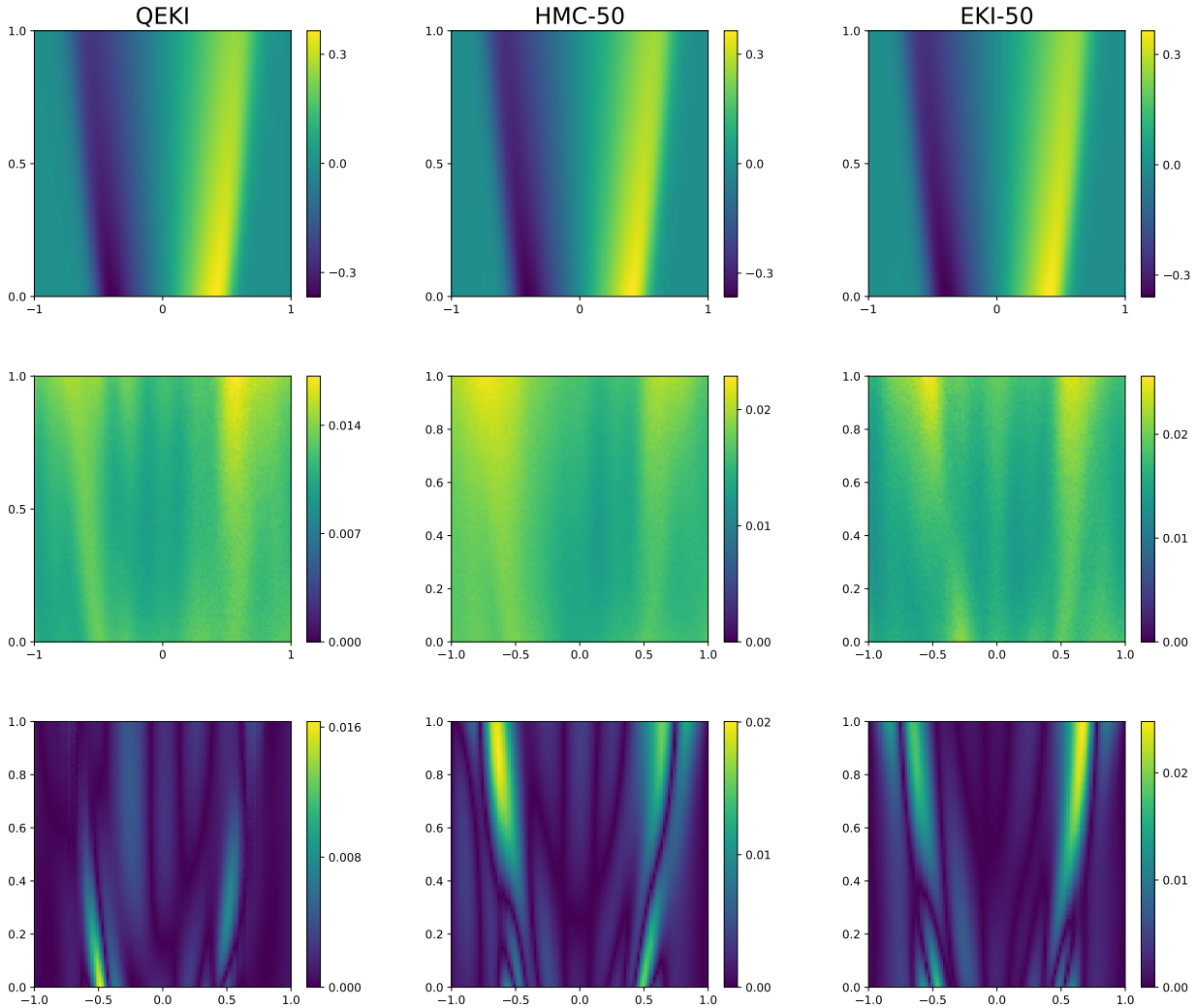


Figure 8. Comparison of QEKI and HMC, EKI with 50-neuron-DNN under noise levels $\sigma_u = 0.01$. The prediction values obtained by these methods (**top row**), the corresponding standard deviations (**middle row**), the absolute error between the ground-truth solution (**bottom row**).

Table 7. Comparison of QEKI and EKI, HMC with classical DNNs of different sizes under noise level $\sigma_u = 0.01$, together with convergence, relative errors e_u and e_v of the unknown parameter v.

	Trainable Parameters	Converged	e_u	e_v
QEKI	163 (+18)	yes	1.41%	2.57%
HMC	152	no	-	-
	1052	no	-	-
	2752	yes	1.68%	2.88%
EKI	152	no	-	-
	1052	no	-	-
	2752	yes	1.72%	2.97%

4. Conclusions

This work introduced QEKI, a hybrid inversion framework that combines Ensemble Kalman Inversion (EKI) with a Quantum Neural Network (QNN) to solve PDE-based inverse problems. Experiments demonstrate that QEKI can achieve the same level of accuracy as classical EKI-based approaches while utilizing significantly fewer trainable parameters. To address current difficulties in training QNNs with gradients, EKI, a gradient-free optimization strategy for QNN parameter updates, demonstrates that reliable inversion can still be achieved without back-propagated quantum gradients.

The study shows promise, but there are still constraints that need to be addressed.

(1) The feasible circuit size is capped by the number of qubits and circuit depth. These two factors set the practical ceiling for what we can simulate today rather than the learning method itself. As the circuit grows, the simulation burden quickly outweighs the benefit of testing larger models.
(2) High-dimensional PDE inverse problems cannot yet be handled directly, since current QNN encoding capacity is limited by available qubits.
(3) The EKI update process can be unstable, and we observed that QEKI inherits this issue, leading to fluctuating convergence behavior in some cases.

Looking ahead, progress can be made even before quantum hardware scales up. A promising direction is to explore more efficient quantum network designs—for example, circuit ansatzes that are smaller, better structured, and tailored to PDE solution patterns. By improving how information and parameters are arranged in the circuit, we hope to expand simulation capacity without relying solely on adding qubits or depth. Second, strengthening the stability of the EKI update, for example, by adaptive ensemble re-scaling, improved noise modeling, or regularized Kalman updates, could lead to more predictable convergence. Finally, to evaluate high-dimensional inverse tasks under qubit constraints, dimensionality reduction techniques, such as KL expansion, VAE, or other latent-space parameterizations, can be incorporated, enabling QEKI to test surrogate inversion of high-dimensional parameters in a lower-dimensional space.

In summary, QEKI demonstrates that quantum models can achieve classical inversion accuracy with fewer parameters, and that QNNs can be optimized without requiring explicit quantum gradients. Future work should focus on improving model scalability, algorithm stability, and testing high-dimensional problems through the use of reduced representations.

Author Contributions: Conceptualization, J.Y. and S.T.; methodology, J.Y. and S.T.; software, J.Y.; validation, J.Y.; formal analysis, J.Y.; writing—original draft preparation, J.Y. and S.T.; writing—review and editing, J.Y. and S.T.; visualization, J.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Data is contained within the article.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Karniadakis, G.E.; Kevrekidis, I.G.; Lu, L.; Perdikaris, P.; Wang, S.; Yang, L. Physics-informed machine learning. *Nat. Rev. Phys.* **2021**, *3*, 422–440. https://doi.org/10.1038/s42254-021-00314-5.
2. Raissi, M.; Perdikaris, P.; Karniadakis, G.E. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *J. Comput. Phys.* **2019**, *378*, 686–707. https://doi.org/10.1016/j.jcp.2018.10.045.
3. Lu, L.; Jin, P.; Karniadakis, G.E. Deeponet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators. *arXiv* **2019**, arXiv:1910.03193.
4. Li, Z.; Kovachki, N.; Azizzadenesheli, K.; Liu, B.; Bhattacharya, K.; Stuart, A.; Anandkumar, A. Fourier neural operator for parametric partial differential equations. *arXiv* **2020**, arXiv:2010.08895.
5. Yang, L.; Meng, X.; Karniadakis, G.E. B-PINNs: Bayesian physics-informed neural networks for forward and inverse PDE problems with noisy data. *J. Comput. Phys.* **2021**, *425*, 109913. https://doi.org/10.1016/j.jcp.2020.109913.
6. Pensoneault, A.; Zhu, X. Efficient Bayesian Physics Informed Neural Networks for inverse problems via Ensemble Kalman Inversion. *J. Comput. Phys.* **2024**, *508*, 113006. https://doi.org/10.1016/j.jcp.2024.113006.
7. Gao, Z.; Karniadakis, G.E. Scalable Bayesian Physics-Informed Kolmogorov-Arnold Networks. *SIAM-ASA J. Uncertain. Quantif.* **2025**, *13*, 1543–1577. https://doi.org/10.1137/25M1729253.
8. Iglesias, M.A.; Law, K.J.H.; Stuart, A.M. Ensemble Kalman methods for inverse problems. *Inverse Probl.* **2013**, *29*, 045001. https://doi.org/10.1088/0266-5611/29/4/045001.
9. Kovachki, N.B.; Stuart, A.M. Ensemble Kalman inversion: A derivative-free technique for machine learning tasks. *Inverse Probl.* **2019**, *35*, 095005. https://doi.org/10.1088/1361-6420/ab1c3a.
10. Schuld, M.; Sinayskiy, I.; Petruccione, F. The quest for a Quantum Neural Network. *Quantum Inf. Process.* **2014**, *13*, 2567–2586. https://doi.org/10.1007/s11128-014-0809-8.
11. Du, Y.; Hsieh, M.H.; Liu, T.; You, S.; Tao, D. Learnability of Quantum Neural Networks. *PRX Quantum* **2021**, *2*, 040337. https://doi.org/10.1103/PRXQuantum.2.040337.
12. Panella, M.; Martinelli, G. Neural networks with quantum architecture and quantum learning. *Int. J. Circ. Theor. App.* **2011**, *39*, 61–77. https://doi.org/10.1002/cta.619.
13. Rigatos, G.; Tzafestas, S. Neurodynamics and attractors in quantum associative memories. *Integr. Comput.-Aided Eng.* **2007**, *14*, 225–242. https://doi.org/10.3233/ICA-2007-14303.
14. Schuld, M.; Sweke, R.; Meyer, J.J. Effect of data encoding on the expressive power of variational quantum-machine-learning models. *Phys. Rev. A* **2021**, *103*, 032430. https://doi.org/10.1103/PhysRevA.103.032430.
15. Xiao, Y.; Yang, L.M.; Shu, C.; Chew, S.C.; Khoo, B.C.; Cui, Y.D.; Liu, Y.Y. Physics-informed quantum neural network for solving forward and inverse problems of partial differential equations. *Phys. Fluids* **2024**, *36*, 097145. https://doi.org/10.1063/5.0226232.
16. Berger, S.; Hosters, N.; Möller, M. Trainable embedding quantum physics informed neural networks for solving nonlinear PDEs. *Sci. Rep.* **2025**, *15*, 18823. https://doi.org/10.1038/s41598-025-02959-z.
17. Trahan, C.; Loveland, M.; Dent, S. Quantum Physics-Informed Neural Networks. *Entropy* **2024**, *26*, 649. https://doi.org/10.3390/e26080649.
18. McClean, J.R.; Boixo, S.; Smelyanskiy, V.N.; Babbush, R.; Neven, H. Barren plateaus in quantum neural network training landscapes. *Nat. Commun.* **2018**, *9*, 4812. https://doi.org/10.1038/s41467-018-07090-4.
19. Larocca, M.; Thanasilp, S.; Wang, S.; Sharma, K.; Biamonte, J.; Coles, P.J.; Cincio, L.; McClean, J.R.; Holmes, Z.; Cerezo, M. Barren plateaus in variational quantum computing. *Nat. Rev. Phys.* **2025**, *7*, 174–189. https://doi.org/10.1038/s42254-025-00813-9.
20. Grant, E.; Wossnig, L.; Ostaszewski, M.; Benedetti, M. An initialization strategy for addressing barren plateaus in parametrized quantum circuits. *Quantum* **2019**, *3*, 214. https://doi.org/10.22331/q-2019-12-09-214.
21. Sack, S.H.; Medina, R.A.; Michailidis, A.A.; Kueng, R.; Serbyn, M. Avoiding Barren Plateaus Using Classical Shadows. *PRX Quantum* **2022**, *3*, 020365. https://doi.org/10.1103/PRXQuantum.3.020365.
22. Patti, T.L.; Najafi, K.; Gao, X.; Yelin, S.F. Entanglement devised barren plateau mitigation. *Phys. Rev. Res.* **2021**, *3*, 033090. https://doi.org/10.1103/PhysRevResearch.3.033090.
23. Nielsen, M.A.; Chuang, I.L. *Quantum Computation and Quantum Information*; Cambridge University Press: Cambridge, UK, 2010.
24. Schuld, M.; Bocharov, A.; Svore, K.M.; Wiebe, N. Circuit-centric quantum classifiers. *Phys. Rev. A* **2020**, *101*, 032308. https://doi.org/10.1103/PhysRevA.101.032308.
25. Farhi, E.; Neven, H. Classification with Quantum Neural Networks on Near Term Processors. *arXiv* **2018**, arXiv:1802.06002.

26. Bergholm, V.; Izaac, J.; Schuld, M.; Gogolin, C.; Ahmed, S.; Ajith, V.; Alam, M.S.; Alonso-Linaje, G.; AkashNarayanan, B.; Asadi, A.; et al. Pennylane: Automatic differentiation of hybrid quantum-classical computations. *arXiv* **2018**, arXiv:1811.04968.
27. Bradbury, J.; Frostig, R.; Hawkins, P.; Johnson, M.J.; Leary, C.; Maclaurin, D.; Necula, G.; Paszke, A.; VanderPlas, J.; Wanderman-Milne, S.; et al. *JAX: Composable Transformations of Python+NumPy Programs*, Version 0.2.5; Google LLC: Mountain View, CA, USA, 2018.
28. Zou, Z.; Meng, X.; Psaros, A.F.; Karniadakis, G.E. NeuralUQ: A Comprehensive Library for Uncertainty Quantification in Neural Differential Equations and Operators. *SIAM Rev.* **2024**, *66*, 161–190. https://doi.org/10.1137/22M1518189.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.