Co-evolutionary Asymmetry vs. Modular Design: Optimization Trade-offs in Quantum Denoising Autoencoders

Abstract

In Quantum Machine Learning (QML), modularity, symmetry-awareness, and pretraining are standard heuristics used to mitigate barren plateaus in deep variational circuits. While effective for symmetric tasks like noiseless data compression, we demonstrate that these intuitions falter in asymmetric tasks such as chaotic time series denoising. This paper investigates the denoising of chaotic Mackey–Glass sequences across a spectrum of Quantum Autoencoder (QAE) architectures, contrasting the performance of co-evolutionary asymmetric designs (Monolith) against multi-stage modular models with pre-trained, synchronized latent spaces (Mirror, Sidekick, and Stacked). Our experiments characterize the "Curse of Asymmetry," where extensively pretraining a decoder on clean data carves a rigid latent manifold that prevents the subsequent encoder from generalizing to high-entropy noise. Consequently, the baseline Monolith QAE – optimizing an unconstrained, independent encoder and decoder – significantly outperforms all pretrained modular variants in both denoising capacity and parameter efficiency. Furthermore, our complexity analysis identifies a structural "instability zone" for modular designs, proving that pre-trained models require deeper circuits merely to overcome their initial structural biases. By mapping these failures, this work redefines the architectural prerequisites for quantum time series processing, establishing that models leveraging co-evolutionary asymmetry supersede the presumed safety of modularity and fixed structural priors.

1 Introduction

The prevailing heuristics of Quantum Machine Learning (QML) are increasingly informed by classical deep learning [34] and high-dimensional machine learning applications, such as in computational chemistry [35, 28] and *de novo* protein design [17]. These principles – namely **modularity**, **symmetry-awareness**, and **pretraining** – serve as the foundation for modern architectural design.

In the context of QML development, these heuristics are often essential for navigating the "barren plateau" phenomenon – regions of the cost landscape where vanishing gradients stall optimization [8]. Current literature suggests that mitigating barren plateaus requires a reduction in model complexity, typically achieved by shrinking the parameter or Hilbert space dimensionality. For example, in variational modeling [29] and ground-state estimation [23], removing structural redundancies through symmetry-informed constraints can conserve quantum resources while simultaneously improving performance. Similarly, modular pretraining – via layerwise learning [41], identity blocks [12], or transfer learning [26] – is used to provide a "warm start" for the global optimization process.

However, in this paper, we demonstrate that for the task of denoising chaotic time series via Quantum Autoencoders (QAEs), these established QML practices can fundamentally break down. We explore a range of denoising QAE architectures through the lens of symmetry and symmetry breaking, revealing that chaotic dynamics pose unique challenges that defy conventional modular and symmetric intuitions.

Note on notation: Following standard quantum mechanical convention, operators in the subsequent equations are applied from right to left; the operator adjacent to the density matrix denotes the initial physical transformation. However, all presented quantum circuit diagrams follow the standard chronological left-to-right flow.

2 Denoising QAEs for Time Series

The recovery of clean sequences from noisy observations is a long-standing problem in control and forecasting [44, 14, 4] and engineering [38, 20], and more recently in financial [40], medical [43], biomedical [42], and scientific [21] applications. Traditional solutions to this problem include linear filters and spectral methods, and highly effective machine learning approaches [3, 2, 19], including deep learning models, such as denoising autoencoders (AE) and generative technologies (such as GANs) [15]. Some of these approaches also extend noise discovery and elimination to anomaly detection [13]. The effectiveness of more sophisticated methods, such as AEs and GANs, often comes at the cost of large models, extensive data augmentation, and careful regularization, especially for non-stationary time series [11, 3, 22]. In resource-constrained regimes, these requirements can result in slow experimentation cycles and difficult training.

Quantum Autoencoders offer a complementary perspective on sequence denoising, where variational quantum algorithms could offer some representational and processing efficiencies [1], such as utilization of the exponentially large feature representation and manipulation in the Hilbert state space and a significant reduction in the number of trainable parameters to achieve denoising performance comparable to that of their classical counterparts.

Quantum Autoencoders (QAE) find their roots in classical Autoencoders (AE) [11, ch14]. There are many applications of QAEs, including data compression [37], image processing [16],

media analysis [36], anomaly detection [39, 27] and support for high-energy physics [31]. In the context of time series analysis, the main objective for denoising AEs and QAEs is lossy compression of temporal data [7, 25], which can be used to remove infrequent or unimportant properties of data sequences [6].

Despite growing interest in QAE, prior work in this area has largely emphasized their application, e.g., to data compression, quantum state preparation, or noise reduction, with limited work devoted to QAE architectural choices aimed at improving their training effectiveness and efficiency of sequence denoising [10]. This paper further addresses this issue.

2.1 General architecture of a denoising QAE

Our research on denoising QAE focuses specifically on models mapping noisy to noise-reduced time series (or sequences), represented as a series of windows (or sub-sequences) of fixed length K sliding with a stride d (also referred to as *step*) along the length of the time series. In the following sections, it will be assumed that quantum circuits of N qubits encode windows of K values, one per qubit, so $K = N$.

2.1.1 Purity and noise in time series data

Let $Z = \{z_t\}_{t=1}^{T}$ be a one-dimensional classical time series, indexed from 1 to T, where $z_t \in \mathbb{R}$. To process these data within a quantum framework, we employ a sliding window mechanism to extract local temporal features.

For a fixed window size K and a stride d, the m-th window of the clean time series is defined as the vector:

$$s^{(m)} = [z_{m\cdot d+1}, z_{m\cdot d+2}, \ldots, z_{m\cdot d+K}]^\top \in \mathbb{R}^K, \tag{1}$$

where $m \in \{0, 1, \ldots, M-1\}$ and $M = \lfloor (T-K)/d \rfloor + 1$. For notational simplicity, we drop the index m and denote a generic clean window as $s \in \mathbb{R}^K$. We assume an additive noise model, where the observed (noisy) window $x \in \mathbb{R}^K$ is given by:

$$x = s + \eta, \tag{2}$$

where s denotes the deterministic underlying sequence (the clean target) and $\eta \in \mathbb{R}^K$ represents a random disturbance, typically modeled as white Gaussian noise $\eta \sim \mathcal{N}(0, \sigma^2 \mathbf{I}_K)$. The denoising objective is to learn a map $x \mapsto \hat{s}(x)$ that reconstructs s from x.

2.1.2 Typical QAE structure

In general, denoising QAEs are commonly composed of two sub-circuits, i.e., a quantum encoder and a quantum decoder, separated by a partially closed barrier consisting of two sets of qubits, i.e. the *latent space* (or "code") that flows (compressed) information from the encoder to the decoder, and *trash space* that prevents information flow by reinitializing its qubits [37, 16, 5] (Figure 1). This explicit latent-trash split is utilized in denoising, with the intention for the pure component of the sequence to be channeled through the latent space for decoding, while the noise component to be diverted into the trash space and discarded.

- Quantum model: a circuit of N qubits to match time series windows of N values (this may vary depending on the QAE objectives).

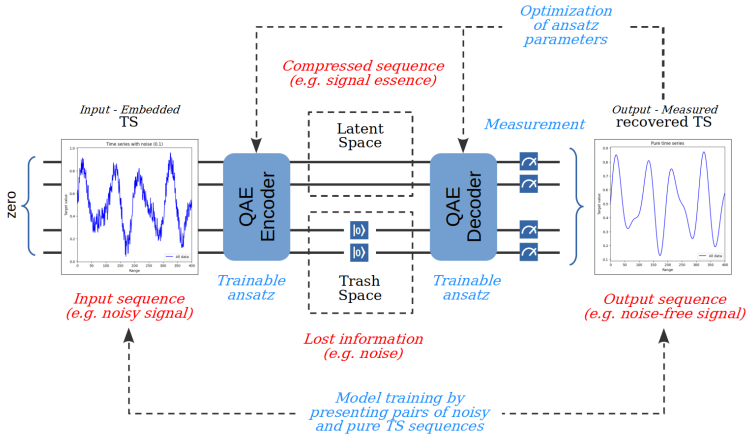


Figure 1: A typical denoising QAE (for time series)

- Input layer ($\rho(x)$): a quantum feature map embedding a window of noisy classical TS values (x) on the input, thus preparing the circuit state ($\rho(x)$). All investigated QAE architectures adopt the same method of angle encoding of data, where individual values of the time series window are embedded in a quantum circuit using single-qubit rotations. However, each QAE architecture assumes a differently scaled range of encoded values.
- QAE encoder (U_E): a quantum ansatz of several layers consisting of trainable parameter blocks and entangling blocks, evolving the state of N qubits into a state of n qubits ($n < N$), therefore compressing input into latent space. In the process of optimization, the encoder learns to separate the essential information, which is compressed and directed into a *latent space* of the circuit, from the superfluous information, which is steered away into a *trash space* of the circuit, where it is destroyed.
- Latent space (H_L): representing the essential features of the window embedded in the initial circuit state (within Hilbert space).
- Trash space (H_T): representing unimportant information, aimed at being removed in QAE training (such as signal noise), to prevent its information flow to the QAE decoder (within Hilbert space). The process of information destruction is achieved by resetting the trash qubits, either by measurement with reinitialization or by swapping their content with zero initialized ancillary qubits (preferred).
- QAE decoder (U_D): a quantum ansatz of several layers consisting of trainable parameter blocks and entangling blocks, evolving the state of the latent space of n qubits into a state of N qubits, therefore, decompressing the latent space into output. The decoding of compressed information flowing through the latent area to allow partial reconstruction of the pure input. Decoder is implemented either by a separate circuit trained independently, or by inverting the quantum encoder [37, 16].
- Output layer ($s = \mathcal{M}(\rho_{out})$): a measurement block that results in classical data, which can be interpreted as a TS window of N values approximating noise-free data (s). Each QAE architecture uses a slightly different measurement and interpretation of the circuit states (ensuring comparability of performance scores).

An important idea of QAEs is that, through the model training process, their latent space provides a useful representation of the training dataset. In geometric terms, the trained QAE encoder should be able to map any data point on input into a lower dimensional latent manifold spanning all learned data point representations and back into its high-dimensional representation. Importantly, the QAE should perform well for previously not seen data points from the population ("generalization") and minimize the mappings into data not existing in the population ("hallucination").

3 Architectures of denoising QAE

We will consider each QAE architecture according to its respective training paradigm.

QAEs can be trained in a variety of ways. The most common is to treat it as a single monolithic model with a large number of parameters optimized simultaneously. This approach to parameter optimization is generally considered inefficient and inviting the emergence of barren plateaus, which impede model training [9]. Other approaches, which are evaluated in this research, aim to modularize the QAE structure into components, each optimized separately using a smaller number of trainable parameters. Such approaches avoid the formation of barren plateaus and are believed to be more effective in model training.

So, we distinguish four distinct architectures of time series denoising QAEs:

- two models that can be trained in a single stage, i.e. (1) a QAE of a monolithic structure (*Monolith* - a single, unified, and indivisible unit); and (2) a QAE of mirror structure, with a decoder (half-QAE) trained and integrated with its own inverse (*Mirror*); and,
- two models that can be trained in two stages, i.e. Stage 1 involving pretraining of a decoder (half-QAE) on clean data, followed by Stage 2 involving encoder training in (3) streamlined (*Stacked*) and (4) parallel (*Sidekick*) architectures on noisy data. After the model integration, the *Stacked* and *Sidekick* QAEs also allow to freeze (*Frozen* variant) or retrain (*Retrained* variant) a selection of their parameters.

All four QAE architectures represent different strategies not only for model construction and training but also for information compression, sequence reconstruction, and dimensionality reduction to aid in the model optimization process. They all have different assumptions as to the symmetries in data and in their architectural components. However, regardless of how they are trained, once their trained components are integrated for execution, their structures are identical, following the general denoising QAE model (see Section 2.1.2).

3.1 QAE symmetry framework

To systematically analyze and compare the proposed QAE architectures, we evaluate them through the lens of symmetry. We distinguish three primary forms of symmetry:

- **Structural symmetry:** The physical layout of the quantum circuit, specifically whether the encoder and decoder are inverse mirror images of one another ($U_E = U_D^\dagger$).
- **Functional symmetry:** Alignment of the computational task and the result, such as mapping a clean distribution back to itself ($s \mapsto s$).
- **Dynamical symmetry:** The symmetrical evolution of the quantum state as it compresses into the latent space and decompresses into the output.

The challenge of sequence denoising inherently requires the intentional breaking of these symmetries to separate deterministic signals from stochastic noise. In our study, symmetry breaking is enforced across several dimensions:

1. **Data symmetry:** Broken by evaluating the models on complex, chaotic time series (such as Mackey-Glass data) rather than simple trigonometric functions that easily decompose into foundational rotational operators.

2. **Input-output symmetry:** Broken by transitioning from a classical autoencoding task ($s \mapsto s$) to a denoising task mapping noisy observations to clean targets ($x \mapsto s$).

3. **QAE function symmetry:** Broken by enforcing a strict informational bottleneck of the latent space. While classical inputs are mapped to an exponentially large Hilbert space, we severely restrict the latent space capacity. Without an active trash space to discard information, leakage cannot occur; therefore, the latent dimension must be restricted enough to force the model to prioritize the signal over the noise.

4. **Latent representation symmetry:** Broken by decoupling the encoder and decoder latent spaces, thus requiring active synchronization during training.

5. **Weight symmetry:** Broken by independent weight updates or retraining stages, decoupling the parameter spaces of the encoder and decoder.

We categorize our four QAE architectures based on how their training strategies preserve or break these specific symmetries.

3.2 Quantum state encoding and partitioning of qubits

To interface classical windows with a quantum circuit, we embed the input vector into an N-qubit state using a feature map $U_{\text{in}}(\cdot)$. To maintain a general framework independent of the specific encoding, we represent the encoded inputs as density matrices:

$$\rho(x) = U_{\text{in}}(x)\,|0\rangle^{\otimes N}\,\langle 0|^{\otimes N}\,U_{\text{in}}^{\dagger}(x), \quad \rho(s) = U_{\text{in}}(s)\,|0\rangle^{\otimes N}\,\langle 0|^{\otimes N}\,U_{\text{in}}^{\dagger}(s). \tag{3}$$

Across all architectures (Figure 2), the N-qubit separation between the QAE encoder and decoder is divided into two subsystems:

- **Latent space ($\mathcal{H}_L$):** consisting of n qubits, intended to store the compressed representation of the input sequence.

- **Trash space ($\mathcal{H}_T$):** consisting of $N - n$ qubits ($n < N$), intended to store noise and redundant information to be discarded.

This induces a Hilbert space decomposition $\mathcal{H} \cong \mathcal{H}_L \otimes \mathcal{H}_T$. The guiding principle is that the reconstruction-relevant information from s should be concentrated in L, while the stochastic component η should be routed to T and driven to a fixed reference state.

Note: To make the notation clearer throughout, we will explicitly annotate the Hilbert subspace constructs with the appropriate reference tag L for *latent* or T for *trash*, e.g. $|0\rangle_T$ rather than a less direct $|0\rangle^{\otimes(N-n)}$. Also, to distinguish the constructs resulting from different stages of model training, we will index them with stage number, e.g. $\rho_{\text{L}}^{(1)}(x;\theta)$ or $\rho_{\text{out}}^{(2)}(x;\theta)$.

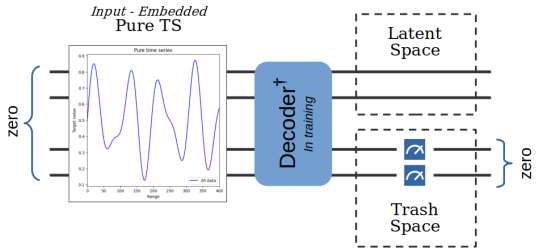


Figure 2: Half-QAE (Inverted Decoder)

3.3 Half-QAE (Inverted Decoder)

Three QAE models *Mirror*, *Stacked* and *Sidekick* require a decoder that is separately pre-trained (Stage 1) in the form of a *Half-QAE* (Figure 2). *Half-QAE* is a unitary circuit that takes a sequence of noise-free data s at the input and splitting it at the output into a canonical *latent* representation of the input and some residual *trash* information.

Let $U_D(\boldsymbol{\phi})$ be a parameterized unitary circuit (the decoder) with a vector of trainable parameters $\boldsymbol{\phi} \in \mathbb{R}^p$. When applied to the noise-free state associated with embedded sequence $\rho(s)$, it prepares the joint latent-trash state:

$$\rho_{LT}^{(1)}(s;\boldsymbol{\phi}) = U_D^\dagger(\boldsymbol{\phi})\rho(s)U_D(\boldsymbol{\phi}). \tag{4}$$

Given a clean input window s, we prepare the state $\rho(s)$ and apply the inverted decoder to obtain the reduced trash state $\rho_T(s;\boldsymbol{\phi})$; simultaneously, the compressed information flows through the latent space, which is "traced out" and ignored during optimization:

$$\rho_T^{(1)}(s;\boldsymbol{\phi}) = \mathrm{Tr}_L\left[\rho_{LT}^{(1)}(s;\boldsymbol{\phi})\right], \quad \rho_L^{(1)}(s;\boldsymbol{\phi}) = \mathrm{Tr}_T\left[\rho_{LT}^{(1)}(s;\boldsymbol{\phi})\right]. \tag{5}$$

In the same way, we obtain $\rho_L(s;\boldsymbol{\phi})$. The objective is to find parameters $\boldsymbol{\phi}^*$ such that the trash space is driven toward the vacuum state $|0\rangle\langle 0|^{\otimes(N-n)}$. This is achieved by maximizing the fidelity (or probability of all-zeros, typically using a SWAP test [33]) on the trash register:

$$\min_{\boldsymbol{\phi}} \mathcal{L}_{half}(\boldsymbol{\phi}) = \mathbb{E}_s\left[1 - \langle 0|_T\, \rho_T^{(1)}(s;\boldsymbol{\phi})\, |0\rangle_T\right]. \tag{6}$$

Note that while $\rho(s)$ provides the "clean" state, the mapping $U_D(\phi)$ effectively learns a manifold for clean dynamics, where the essential aspects of s are captured by the n latent qubits. However, the trained unitary is "blind" to the noise, which will be introduced in the later stage of the training process.

The *half-QAE* training process relies on two features of quantum unitaries:

- **Reversibility of unitaries**, i.e. all unitary quantum operations are invertible [32, ch 3.2.5], therefore, the reversed half-QAE unitary can be used without any changes as a decoder (its result may need to be reversed as well).
- **Preservation of quantum information**, i.e. any information not reaching the trash space must have been redirected to the latent space (this feature of unitaries is implied by their property of reversibility, i.e. for any unitary U we get $U^\dagger U = \mathbf{I}$).

When using *half-QAE* as part of the pretraining of the decoder (Stage 1), the QAE encoder $U_E(\boldsymbol{\phi}^*) \equiv U_D^\dagger(\boldsymbol{\phi}^*)$ together with the decoder $U_D(\phi^*)$ can be integrated to form the two key components of the QAE model ready for further processing or training (Stage 2), as required by their respective architectures.

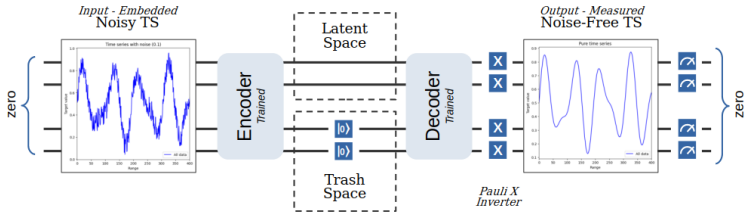


Figure 3: *Mirror* QAE (note the need to also invert the result)

3.4 Mirror QAE Architecture

What we call the *Mirror* is one of the better-known QAE architectures. A *Mirror* QAE is composed of two identical unitaries (including weights) joined into a mirror-like structure (Figure 3). The *Mirror* encoder is essentially an inverted decoder $U_E(\boldsymbol{\phi}) \equiv U_D^\dagger(\boldsymbol{\phi})$. The main application of *Mirror* QAE is lossy data compression [37]. However, it can also be used to denoise data [30]. Its most interesting feature is its ability to achieve high performance after optimizing only half (encoder or decoder) of its architecture [45, 30, 37].

In our project, the *Mirror* model trains only half of its parameters, i.e. those associated with its decoder $U_D(\boldsymbol{\phi})$ using the *Half-QAE* method (as per Section 3.3). This results in the optimum decoder parameters $\boldsymbol{\phi}^*$.

The *Mirror* final state can therefore be given by:

$$\rho_{\text{out}}^{(2)}(x;\boldsymbol{\theta}) = U_D(\boldsymbol{\theta})\Big(\rho_L^{(1)}(x;\boldsymbol{\theta}) \otimes |0\rangle\langle 0|_T\Big)U_D^\dagger(\boldsymbol{\theta}), \tag{7}$$

where $\rho_L(x;\boldsymbol{\theta}) = \text{Tr}_T\big[U_E(\boldsymbol{\theta})\rho(x)U_E^\dagger(\boldsymbol{\theta})\big]$ and $U_E(\boldsymbol{\theta}) \equiv U_D(\boldsymbol{\theta})^\dagger$, giving the full expansion as:

$$\rho_{\text{out}}^{(2)}(x;\boldsymbol{\theta}) = U_D(\boldsymbol{\theta})\left[\text{Tr}_T\big[U_D(\boldsymbol{\theta})^\dagger\rho(x)U_D(\boldsymbol{\theta})\big] \otimes |0\rangle\langle 0|_T\right]U_D^\dagger(\boldsymbol{\theta}). \tag{8}$$

and $\boldsymbol{\theta} = \boldsymbol{\phi}^*$.

This model is a precursor to the *Stacked* models.

Symmetry profile. The *Mirror* architecture represents the baseline of maximal symmetry. Structurally, it is perfectly symmetric, as the encoder is strictly defined as the adjoint of the decoder ($U_E(\boldsymbol{\phi}) \equiv U_D^\dagger(\boldsymbol{\phi})$). Functionally, during its Stage 1 pre-training, it preserves input-output symmetry by mapping clean sequences to clean sequences. Its symmetry is only broken at inference time when noisy data is introduced ($x \mapsto s$). However, the model itself has no distinct, noise-adapted parameters to handle this asymmetry.

3.5 Stacked Architecture

To improve training stability and compression quality, we developed the Stacked architecture. In this approach, the full QAE circuit is decomposed into two independent components: an encoder U_E and a decoder U_D. The *Stacked* training protocol follows a two-stage optimization procedure, which decouples the discovery of the signal's internal structure from the adaptation to the noise distribution.

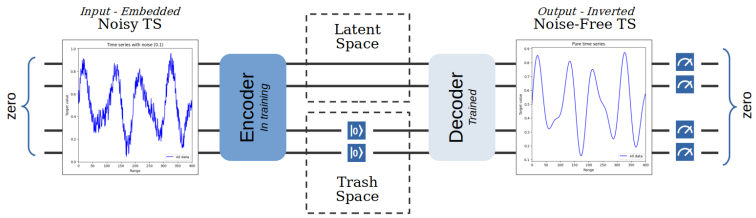


Figure 4: Stage 2 training of the *Stacked* QAE

Stage 1: Learning a Clean Prior. In the first stage, the decoder $U_D(\boldsymbol{\phi})$ is optimized using only noise-free windows s via the *Half-QAE* method (Section 3.3). This yields the optimal decoder weights $\boldsymbol{\phi}^*$, which represent a generative model of the clean signal manifold.

Stage 2: Noise Adaptation with Frozen Decoder. In stage 2 (Figure 4), we introduce the encoder $U_E(\boldsymbol{\theta})$, initialized such that $U_E(\boldsymbol{\phi}^*) = U_D^\dagger(\boldsymbol{\phi}^*)$. During denoising, the noisy input x is compressed into a latent state, the trash qubits are reset, and the state is decoded. The output state is:

$$\rho_{\text{out}}^{(2)}(x;\boldsymbol{\theta},\boldsymbol{\phi}) = U_D(\boldsymbol{\phi})\Big(\text{Tr}_T\left[U_E(\boldsymbol{\theta})\rho(x)U_E^\dagger(\boldsymbol{\theta})\right] \otimes |0\rangle\langle 0|_T\Big)U_D^\dagger(\boldsymbol{\phi}) \tag{9}$$

In the *Stacked (frozen)* variant, we freeze the decoder at $\boldsymbol{\phi}^*$ and optimize only the encoder:

$$\min_{\boldsymbol{\theta}} \mathcal{L}_{Stacked}^{\text{froz}}(\boldsymbol{\theta}) = \mathbb{E}_{(x,s)}\left[\|\mathcal{M}\big(\rho_{\text{out}}^{(2)}(x;\boldsymbol{\theta},\boldsymbol{\phi}^*)\big) - s\|_2^2\right] \tag{10}$$

This ensures the decoder remains a "pure" reconstructor of clean signals while the encoder learns the specific mapping from the noisy distribution to the clean latent space.

Stage 2 Alternative: Retrained Decoder. Alternatively, in the *Stacked (retrained)* variant, both $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ are fine-tuned jointly, using $\boldsymbol{\phi}^*$ as a high-quality "warm start":

$$\min_{\boldsymbol{\theta},\,\boldsymbol{\phi}} \mathcal{L}_{Stacked}^{\text{retr}}(\boldsymbol{\theta},\boldsymbol{\phi}) = \mathbb{E}_{(x,s)}\left[\|\mathcal{M}\big(\rho_{\text{out}}^{(2)}(x;\boldsymbol{\theta},\boldsymbol{\phi})\big) - s\|_2^2\right] \tag{11}$$

Symmetry profile. The *Stacked* architecture maintains structural symmetry at initialization ($U_E = U_D^\dagger$) but intentionally breaks functional and weight symmetries during Stage 2. By fixing the pre-trained decoder in the *Stacked (frozen)* variant and introducing noisy data, the input-output symmetry is severed. The model forces the encoder to break the noise symmetry, compressing the stochastic input into a clean latent representation. In the *Stacked (retrained)* variant, the weight symmetry is also completely broken as $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ diverge to optimize the end-to-end outcome.

3.6 Sidekick Architecture

The most complex design investigated in this work is the *Sidekick* architecture, which optimizes the QAE model in two stages, i.e., Stage 1 to train the decoder and Stage 2 to train the encoder. However, in addition to the encoder and decoder components, it also introduces a third module responsible for the synchronization of their latent spaces in training.

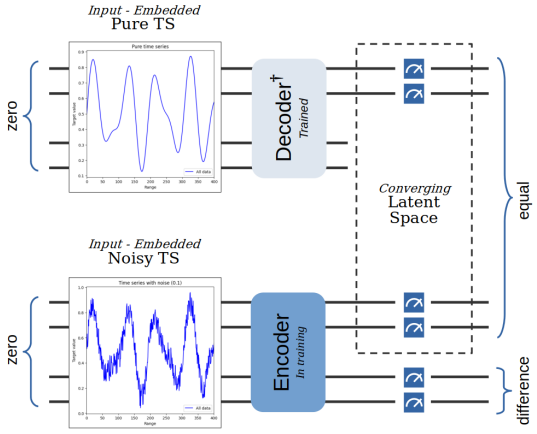


Figure 5: Stage 2 training of the "Sidekick" QAE

On the surface, both the *Stacked* and *Sidekick* architectures are conceptually similar. They both employ a multi-stage training regime to avoid barren plateaus. Both rely on a latent space bottleneck $\mathcal{H}_L$ to act as the primary conduit for information flow. However, their internal mechanics for achieving noise-resilience are fundamentally different.

Stage 1: Learning a Clean Prior. In the first stage, the decoder $U_D(\boldsymbol{\phi})$ is optimized using only noise-free windows s via the *Half-QAE* method (Section 3.3). This yields the optimal decoder weights $\boldsymbol{\phi}^*$, which represent a generative model of the clean signal manifold.

Stage 2: Noise Adaptation with Frozen Decoder. In this stage of training (Figure 5), we introduce the encoder $U_E(\boldsymbol{\theta})$, which is architecturally identical to the inverted decoder $U_E(\boldsymbol{\phi}) = U_D^\dagger(\boldsymbol{\phi})$.

In the *Sidekick (frozen)* variant's training phase, we initialize $\boldsymbol{\theta} = \boldsymbol{\phi}^*$. The objective is to optimize the encoder parameters $\boldsymbol{\theta}$ to minimize the divergence $\mathcal{D}$ between the noisy latent representation and the clean "anchor" representation generated by the fixed decoder:

$$\min_{\boldsymbol{\theta}} \mathcal{L}_{\text{Sidekick}}^{froz} = \mathcal{D}\left(\sigma_L^{(1)}(\boldsymbol{\phi}^*) \parallel \tilde{\sigma}_L^{(2)}(\boldsymbol{\theta})\right) \tag{12}$$

The divergence $\mathcal{D}$ represents the synchronization of the two latent spaces. While this is typically implemented via a SWAP test to maximize state fidelity, it abstractly represents any measure of distance (e.g., Trace distance) between the density matrices. The latent mappings are defined as:

- **Clean Latent Anchor:** $\sigma_L^{(1)}(\boldsymbol{\phi}^*) = \text{Tr}_T\left[U_D^\dagger(\boldsymbol{\phi}^*)\rho(s)U_D(\boldsymbol{\phi}^*)\right]$
- **Noisy Latent Representation:** $\tilde{\sigma}_L^{(2)}(\boldsymbol{\theta}) = \text{Tr}_T\left[U_D^\dagger(\boldsymbol{\theta})\rho(x)U_D(\boldsymbol{\theta})\right]$

The final denoised output $\rho_{\text{out}}^{(2)}(x)$ for a noisy input window $x = s + \eta$ is constructed by composing the optimized encoder $U_D^\dagger(\boldsymbol{\theta}^*)$ with the pre-trained decoder $U_D(\boldsymbol{\phi}^*)$ in a typical QAE pipeline (Figure 1). The full expansion of the *Sidekick (frozen)* output state is:

$$\rho_{\text{out}}^{(2)}(x) = U_D(\boldsymbol{\phi}^*)\left(\text{Tr}_T\left[U_D^\dagger(\boldsymbol{\theta}^*)\rho(x)U_D(\boldsymbol{\theta}^*)\right] \otimes |0\rangle\langle 0|_T\right) U_D^\dagger(\boldsymbol{\phi}^*) \tag{13}$$

Stage 2 Alternative: Retrained Decoder. In the adaptive *Sidekick* configuration, called *Sidekick (retrained)*, the system is initialized with Stage 1 optima, $\boldsymbol{\theta} = \boldsymbol{\phi}^*$ and $\boldsymbol{\phi} = \boldsymbol{\phi}^*$. Then both parameter sets are jointly updated to minimize combined synchronization and leakage loss. We denote the resulting optimal parameters for this regime as $\boldsymbol{\theta}'$ and $\boldsymbol{\phi}'$:

$$\{\boldsymbol{\theta}', \boldsymbol{\phi}'\} = \arg\min_{\boldsymbol{\theta},\boldsymbol{\phi}} \left[\mathcal{L}^{retr}_{\text{Sidekick}}(\boldsymbol{\theta}, \boldsymbol{\phi}) + \lambda\, \mathcal{L}^{retr}_{\text{Leak}}(\boldsymbol{\theta}, \boldsymbol{\phi})\right] \tag{14}$$

where λ represents a regularization penalty that balances the contribution of loss from divergence of components and their information leakage to the thrash, which are defined as follows:

- **Synchronization Loss:** $\mathcal{L}^{retr}_{\text{Sidekick}}$ aligns the noisy latent state with the evolving sequence manifold:

$$\mathcal{L}^{retr}_{\text{Sidekick}}(\boldsymbol{\theta}, \boldsymbol{\phi}) = \mathcal{D}\left(\text{Tr}_T[U_D^\dagger(\boldsymbol{\phi})\rho(s)U_D(\boldsymbol{\phi})] \parallel \text{Tr}_T[U_D^\dagger(\boldsymbol{\theta})\rho(x)U_D(\boldsymbol{\theta})]\right) \tag{15}$$

- **Leakage Regularization:** $\mathcal{L}^{retr}_{\text{leak}}$ penalizes the presence of signal in the trash subspaces $\mathcal{H}_T$ of both the encoder and decoder:

$$\mathcal{L}^{retr}_{\text{Leak}}(\boldsymbol{\theta}, \boldsymbol{\phi}) = \left(1 - \langle 0|\tilde{\sigma}^{(2)}_T(\boldsymbol{\theta})|0\rangle_T\right) + \left(1 - \langle 0|\sigma^{(1)}_T(\boldsymbol{\phi})|0\rangle_T\right) \tag{16}$$

  and $\sigma^{(1)}_T(\boldsymbol{\phi})$ and $\tilde{\sigma}^{(2)}_T(\boldsymbol{\theta})$ are the reduced density matrices of the trash subspaces for the clean and noisy streams, respectively.

For the retrained regime *Sidekick (retrained)*, the final denoised output $\rho^{(2)}_{\text{out}}(x)$ utilizes the refined parameter sets to map the noisy input through the synchronized bottleneck:

$$\rho^{(2)}_{\text{out}}(x) = U_D(\boldsymbol{\phi}')\left(\text{Tr}_T\left[U_D^\dagger(\boldsymbol{\theta}')\rho(x)U_D(\boldsymbol{\theta}')\right] \otimes |0\rangle\langle 0|_T\right) U_D^\dagger(\boldsymbol{\phi}') \tag{17}$$

Comparison of Sidekick vs. Stacked. The conceptual similarity between the *Stacked* and *Sidekick* models – namely two-stage training and the synchronization of a encoder/decoder latent spaces to ensure the smooth flow of information – masks a fundamental difference in their learning objectives:

- **Outcome-Oriented Reconstruction (Stacked):** The *Stacked* model optimizes its parameters by directly comparing the final denoised outcome ρ_{out} with the clean ground-truth signal s. In Stage 2 training, the latent space is a functional bottleneck that must be traversed to minimize end-to-end reconstruction error.
- **Manifold-Oriented Convergence (Sidekick):** In contrast, the *Sidekick* model focuses entirely on latent synchronization within the Hilbert space. During Stage 2 training, the model never explicitly evaluates the final output against the clean data. Instead, it seeks to converge the encoder's mapping of noisy data, $U_D^\dagger(\boldsymbol{\theta})\rho(x)U_D(\boldsymbol{\theta})$, toward the pre-trained decoder's representation of clean data, $U_D^\dagger(\boldsymbol{\phi}^*)\rho(s)U_D(\boldsymbol{\phi}^*)$.

By enforcing this internal latent alignment, the *Sidekick* architecture treats denoising not as a corrective transformation of the output but as a synchronization of sequence manifolds, where the encoder learns to identify the patterns of a clean sequence within the stochastic noise.

Symmetry profile. The *Sidekick* architecture introduces the most complex symmetry dynamics. Although it maintains structural symmetry ($U_E = U_D^\dagger$), it explicitly breaks the symmetry of the latent representation. Because clean and noisy sequences are processed by functionally decoupled unitaries during Stage 2, their respective latent spaces exist in isolation. The primary objective of *Sidekick* is to computationally force the restoration of dynamical symmetry – synchronizing the noisy latent representation with the clean anchor – despite the severe input-output asymmetry.

Note: The fidelity of the encoder/decoder latent spaces serves as an indirect proxy metric for the cost function.

3.7 Monolith Architecture

The *Monolith* (Figure 1) is the most general form of a denoising QAE, where the encoder and decoder may have entirely different structures; however, it has been common to assume that the encoder is the adjoint of the decoder. In this architecture, the encoder and decoder are treated as a single, jointly trainable unit $(\mathcal{E}_{\boldsymbol{\theta}}, \mathcal{D}_{\boldsymbol{\phi}})$.

Encoder and Latent-Trash Decomposition. Let $U_E(\boldsymbol{\theta})$ be a parameterized unitary circuit (the encoder) with a vector of trainable parameters $\boldsymbol{\theta} \in \mathbb{R}^p$. When applied to the encoded noisy state $\rho(x)$, it prepares the joint latent-trash state:

$$\rho_{LT}(x;\boldsymbol{\theta}) = U_E(\boldsymbol{\theta})\rho(x)U_E^\dagger(\boldsymbol{\theta}). \tag{18}$$

The encoder's objective is to find a transformation that concentrates the reconstruction-relevant features of s into the reduced latent state $\rho_L(x;\boldsymbol{\theta})$; simultaneously, the encoder is encouraged to route the redundant and noisy components of the input into the trash subsystem $\rho_T(x;\boldsymbol{\theta})$:

$$\rho_L(x;\boldsymbol{\theta}) = \mathrm{Tr}_T\big[\rho_{LT}(x;\boldsymbol{\theta})\big], \quad \rho_T(x;\boldsymbol{\theta}) = \mathrm{Tr}_L\big[\rho_{LT}(x;\boldsymbol{\theta})\big]. \tag{19}$$

The Information Bottleneck (Trash Reset). The defining feature of the *Monolith* model is the explicit removal of information stored in the trash space before the decoding stage. This is modeled as a quantum channel $\mathcal{R}_T$ that maps any state in $\mathcal{H}_T$ to the fixed vacuum state $|0\rangle\langle 0|^{\otimes(N-n)}$. The resulting post-reset state is a product state:

$$\rho'_{LT}(x;\boldsymbol{\theta}) = (\mathcal{I}_L \otimes \mathcal{R}_T)\big(\rho_{LT}(x;\boldsymbol{\theta})\big) = \rho_L(x;\boldsymbol{\theta}) \otimes |0\rangle\langle 0|_T, \tag{20}$$

where $\mathcal{I}_L$ denotes the identity map on the latent space. Because the underlying sequence s is structured and compressible, while the noise η is high-entropy and stochastic, this bottleneck forces the network to prioritize the preservation of s over η.

Decoder, Readout, and Objective. The decoder $U_D(\boldsymbol{\phi})$, parametrized by $\boldsymbol{\phi} \in \mathbb{R}^q$ (where $\boldsymbol{\phi}$ is independent of $\boldsymbol{\theta}$), acts on the purified state to prepare the output density matrix:

$$\rho_{\text{out}}(x;\boldsymbol{\theta},\boldsymbol{\phi}) = U_D(\boldsymbol{\phi})\rho'_{LT}(x;\boldsymbol{\theta})U_D^\dagger(\boldsymbol{\phi}). \tag{21}$$

which fully expands to:

$$\rho_{\text{out}}(x;\boldsymbol{\theta},\boldsymbol{\phi}) = U_D(\boldsymbol{\phi})\left[\mathrm{Tr}_T\big[U_E(\boldsymbol{\theta})\rho(x)U_E^\dagger(\boldsymbol{\theta})\big] \otimes |0\rangle\langle 0|_T\right] U_D^\dagger(\boldsymbol{\phi}). \tag{22}$$

The classical denoised window prediction $\hat{s}(x;\boldsymbol{\theta},\boldsymbol{\phi}) \in \mathbb{R}^K$ is obtained by a differentiable measurement operator $\mathcal{M}$:

$$\hat{s}(x;\boldsymbol{\theta},\boldsymbol{\phi}) = \mathcal{M}\big(\rho_{\text{out}}(x;\boldsymbol{\theta},\boldsymbol{\phi})\big). \tag{23}$$

The *Monolith* model is optimized in a supervised manner by minimizing the Mean Squared Error (MSE) between the prediction and the clean target s over the training set:

$$\min_{\boldsymbol{\theta},\boldsymbol{\phi}} \mathcal{L}_{\text{MSE}}(\boldsymbol{\theta},\boldsymbol{\phi}) = \mathbb{E}_{(x,s)}\left[\|\hat{s}(x;\boldsymbol{\theta},\boldsymbol{\phi}) - s\|_2^2\right]. \tag{24}$$

Symmetry profile. The *Monolith* architecture represents total symmetry breaking. Structurally, there is no constraint that enforces an inverse relationship between the encoder and the decoder ($U_E \neq U_D^\dagger$), and their weights $(\boldsymbol{\theta},\boldsymbol{\phi})$ are initialized and optimized entirely independently, in a co-evolutionary asymmetry. The *Monolith* lacks the functional symmetry of a clean pre-training stage. Instead, it relies entirely on the functional asymmetry, i.e. utilizing the $\mathcal{R}_T$ trash reset bottleneck and MSE loss to aggressively force the separation of the deterministic sequence from stochastic noise in a single training stage.

4 Experimental design

Data generation. The sequence data used in our experiments was synthetic, generated using the Mackey–Glass delay differential equation [24]. This system is a standard benchmark for chaotic time series and denoising [18]. We employ the normalized form ($\theta = 1$) of the Mackey-Glass equation:

$$\frac{dx(t)}{dt} = \frac{\beta\, x(t-\tau)}{1 + (x(t-\tau)/\theta)^n} \;-\; \gamma\, x(t), \tag{25}$$

where $x(t)$ is the state variable, τ is the time delay, β controls the strength of nonlinearity, n is the slope of nonlinearity and γ is the decay rate. Following convention, the half-saturation constant θ is set to 1.

To ensure that the captured sequence reflected a mature chaotic regime, we initially generated a series of length 1200 using the **Forward Euler** method with a discretization step of $\Delta t = 1$. Subsequently, this sequence was downsampled by a factor of 6 to a final length of 200 values. This approach preserves the essential temporal features and variety of the chaotic attractor while maintaining a computationally manageable sample size for quantum simulation.

Selection of Mackey-Glass parameters. Two distinct data sets were generated to evaluate model performance across different chaotic regimes. These sets differed only by the delay parameter τ, while all other parameters remained identical: $\beta = 0.2$, $\gamma = 0.1$, $n = 10$, and initial condition $x_0 = 1.2$.

- **Dataset 1 ($\tau = 17$):** Produced more regular oscillations with changes occurring at higher frequencies.
- **Dataset 2 ($\tau = 30$):** Displayed significantly more chaotic behavior with changes occurring across a broader frequency spectrum.

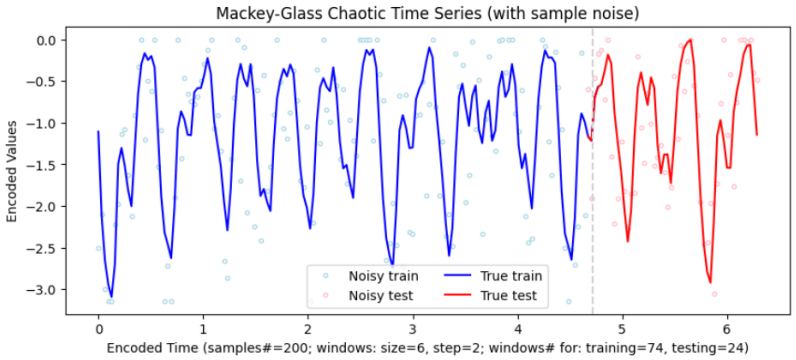


Figure 6: Sample Mackey-Glass Sequence (Flat, MG: $\tau = 30$) with added noise (σ=0.2, clipped). MG sequence was then split into overlapping windows (of varying size and stride)

In this paper, we focus mainly on the results from the analysis of the more complex data $\tau = 30$ (Figure 6).

The generated time series values were scaled to the specific input ranges required by each QAE data encoding scheme. To prevent the models from over-fitting to specific noise artifacts, noise was resampled dynamically during each training iteration (while the underlying Mackey-Glass sequence remained fixed) and clipped to the maximum range of the time series values

Data preparation. We adopted a sliding window approach for time series representation. The *window size* was $W \in \{4, 6\}$ and *stride* was 2. We set the number of qubits equal to the window size $n_{\text{qubits}} = W$, so that each window component is encoded on a dedicated qubit. The window size $W = 4$ emphasizes local dynamics and shallow circuits, while $W = 6$ tests the scalability and higher expressivity of the model. The stride of 2 produces overlapping windows, which increases the sample count. at the cost of higher inter-window correlation. We utilized a chronological split to avoid data leakage: the first 75% of windows (74 windows for W=6) formed the training set, and the remaining 25% (24 windows) served as the test set. Shuffling was not applied to preserve the temporal sequence of the windows.

As the windows overlap ($stride < W$), the resulting QAE reconstructions also overlap. The final time series reconstruction is achieved by averaging the overlapping values, providing a smoothing effect as depicted in Figure 6.

Experimental framework. This study focused on a comparative examination of denoising performance of the QAEs described above, i.e. *Monolith*, *Mirror*, *Stacked*, and *Sidekick*, plus the *Frozen* and *Retrained* variants of Stacked and Sidekick QAEs. All quantum models were built and tested using PennyLane state vector simulation (using "lightning qubits", "autograd" interface and "adjoint" differentiation method). The classical or hybrid equivalents of these models were not investigated (out of scope).

For each QAE architecture, we created several model variants using a wide range of configurations, including 4- and 6-qubit circuits with varying splits between latent and trash qubit ratios (e.g. $3:1, 2:2, 2:4, 4:2$) and the number of circuit layers $L \in \{1, \ldots, 5\}$ layers. The circuits of all QAEs included hardware-efficient ansatze with layers of trainable R_X, R_Y, R_Z gates and ring-style CNOT connectivity. To maintain unitarity and enable efficient adjoint differentiation, we utilized a **SWAP reset** to switch the state of the trash qubits with clean

Table 1: Training/testing consistency for all model variants and training regimes.

Model variant	***N***	***Mean*** Δ_{train} (%)	***Mean*** Δ_{test} (%)	***SD*** ($\Delta_{train} - \Delta_{test}$)	***r*** *(Pearson)*
Monolith*	35	+45.71	+53.70	4.24	0.9946
Sidekick (frozen)	42	+33.61	+44.46	7.30	0.6854
Sidekick (retrained)	35	+11.23	+21.86	24.63	0.7803
Mirror	35	+28.97	+32.78	7.09	0.7859
Stacked (frozen)	42	+43.19	+46.98	3.60	0.8839
Stacked (retrained)	35	+55.54	+61.73	3.46	0.4486

* *As this model training had no variations in its training regime, to calculate statistics we used its structural variants instead.*

ancillas prepared in $|0\rangle$.

Training utilized the Adam optimizer with dynamic learning rates and mini-batching. Pre-training for half-QAE components (in *Mirror*, *Stacked* and *Sidekick* models) used a fidelity-based cost function via the SWAP test. *Siekick* Stage 2 training was also fidelity-based. The remaining integrated QAEs used Mean Square Error (MSE) in Stage 2 training. To ensure robustness against the stochastic nature of variational optimization, results were summarized using the **median** performance across seven pseudo-randomly initialized instances for each configuration. Medians were preferred over means to provide a robust measure of central tendency against occasional convergence failures typical in variational circuits.

5 Results and discussion

Given the computational complexity of quantum simulations and the need to maintain a rigorous separation between training and evaluation, we adopted a training-centric weight selection strategy. Optimal parameters for Stage 1 (compression of clean data) and Stage 2 (denoising) were determined exclusively using training-set metrics. Specifically, Stage 1 weights (Decoder) were selected at the point of maximum fidelity of the SWAP-test in the trash register, while Stage 2 weights (Encoder) were selected at the point of minimum post-training MSE.

To validate that this heuristic does not lead to overfitting or suboptimal generalization, we analyzed the Pearson correlation (r) between training and noise removal performance from the test-set. As shown in Table 1, we observed a high degree of consistency between training and testing for the *Monolith* ($r = 0.99$), *Stacked (frozen)* ($r = 0.88$) and *Mirror* ($r = 0.79$) architectures. *Sidekick (frozen)* ($r = 0.69$) and *Sidekick (retrained)* ($r = 0.79$) were also high.

Although the *Stacked (retrained)* model achieved the highest average denoising performance ($\Delta_{test} = 61.73\%$, see Table 1), we observed a notable decrease in the Pearson correlation coefficient (r=0.45), compared to models with suboptimal performance, such as *Sidekick (retrained)* ($\Delta_{test} = 21.86\%$, $r = 0.78$). However, this decrease in linearity does not indicate failure in model stability; instead, it reflects a performance saturation effect (as illustrated in Figure 7). The results of the *Stacked (retrained)* model are tightly clustered in the high-accuracy regime, consistently outperforming the training expectations (results above the diagonal). In this dense distribution, the reduced range of variance (high precision) mathematically limits the Pearson r value, as the marginal differences in training success no longer serve as sensitive predictors for the already optimized testing outcomes. In contrast, the higher correlation in the *Sidekick*

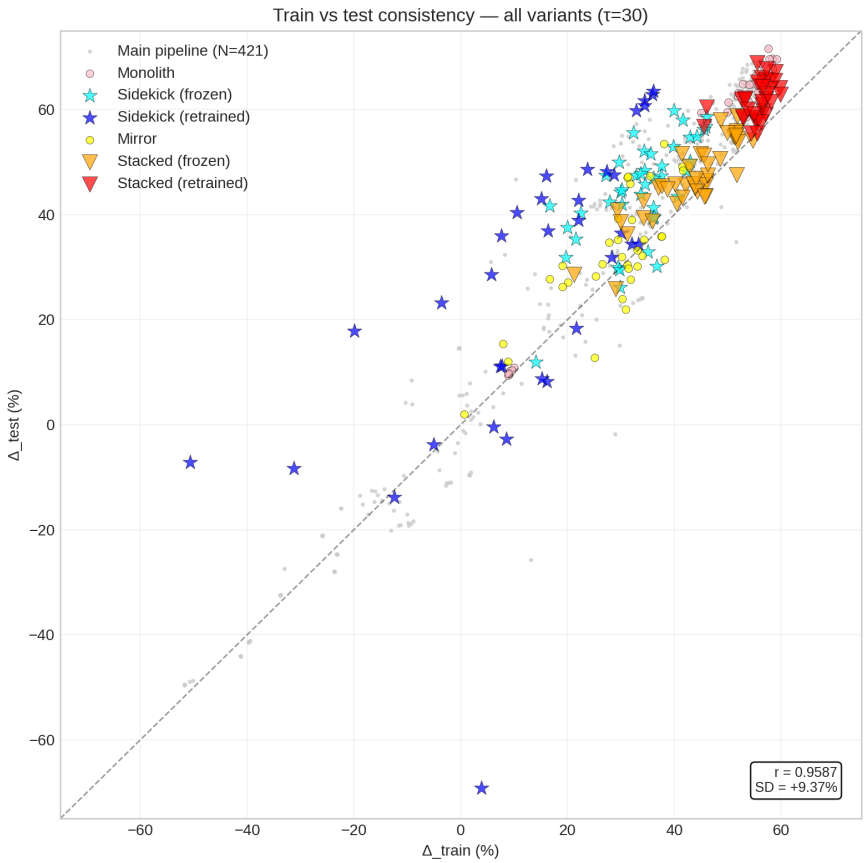


Figure 7: Explanation of the Stacked (retrained) model stability, despite its low Pearson "r". The "main pipeline" refers to other QAE configurations, not considered in this analysis.

model is a byproduct of the wider spread ($SD = 24.63\%$) of its results throughout the performance spectrum, allowing for a more defined linear trend, although of low performance ($M = 21.86\%$ denoising, see Table 1 and Figure 7).

5.1 Analysis of Model Complexity

As noisy intermediate-scale quantum solutions are highly sensitive to coherence times and gate noise, identifying the minimum model complexity required for robust quantum denoising was deemed critical. Therefore, we analyzed the complexity of QAEs in two dimensions: the number of circuit layers (relating to its depth - impacting coherence time) and representational capacity (number of trainable parameters - sensitivity to gate noise) (Figures 8 and 9).

Circuit Layers and the Stability Zone. We first examined the distribution of denoising test performance as a function of the number of circuit layers (1 through 5). In this analysis, we focussed on the *frozen* models, as these were also the basis for the models with *retraining*. We excluded the *Mirror* model as the model is a precursor to the *Stacked* models.

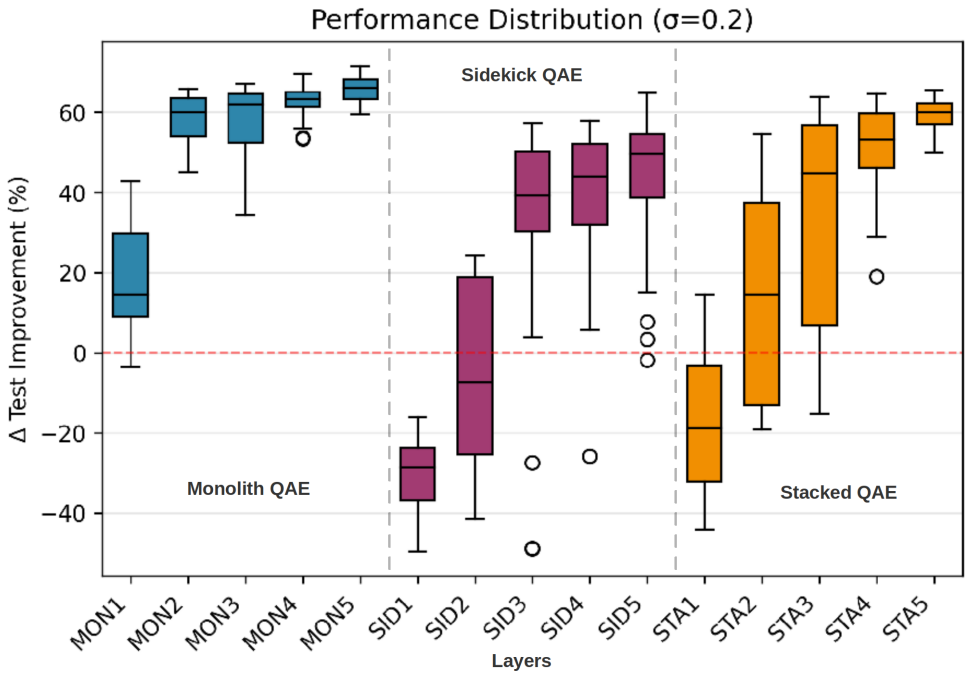


Figure 8: Impact of the number of layers on the QAE denoising performance (*frozen* models, MG $\tau = 30$)

As shown in the performance distribution (Figure 8), the *Sidekick* and *Stacked* models of 1 and 2 layers lacked the expressive power to effectively map noisy and chaotic Mackey-Glass ($\tau = 30$) inputs to their clean targets, often resulting in errors higher than the level of injected noise (negative test improvements).

At fewer layers (1 to 3), the multi-stage architectures (*Sidekick* and *Stacked*) exhibited extreme performance variance. The models possessed just enough capacity to attempt the denoising task, but remained highly sensitive to weight initialization and the constraints of Stage 1 pretraining. The denoising variations reduced for all model architectures above 4 layers of depth, which we identify as the "stability zone," and which was consequently selected for their direct comparison.

The *Sidekick* architecture demonstrated a slightly improved performance distribution at deeper layers; however, the strategy of matching latent representations seems to limit its further learning. In contrast, the *Monolith* model achieves high and stable performance almost immediately at 2 layers, bypassing the instability zone entirely. The *Stacked* models converged just below the *Monolith* performance level.

Parameter Efficiency and Capacity Saturation. Although increased circuit depth stabilizes the variance of multi-stage architectures, evaluating their denoising performance against their complexity, expressed in terms of the number of trainable parameters (Figure 9), reveals hard limits on their models ultimate representational capacity. We mapped the denoising performance against parameter counts ranging from 20 to 180 to identify the average saturation points of each architecture.

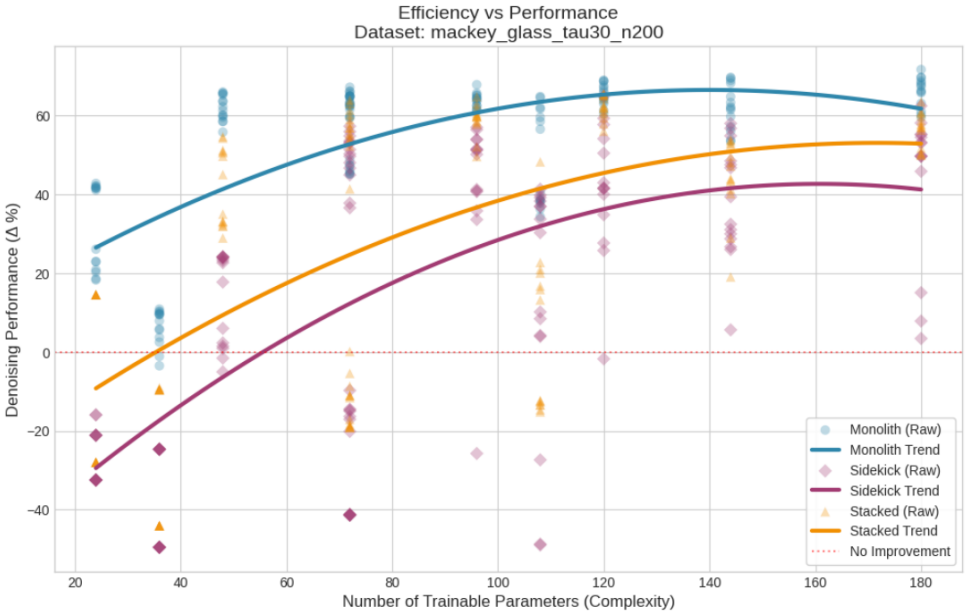


Figure 9: Impact of the number of circuit parameters on denoising performance (Frozen models, MG $\tau = 30$)

The unconstrained *Monolith* architecture demonstrates optimal parameter efficiency (on average). Its performance trajectory rises sharply and saturates at approximately 140 trainable parameters, achieving the global peak for this dataset before showing signs of modest overfitting. This indicates that 140 parameters provide the exact degrees of freedom required to capture the underlying sequence dynamics without memorizing the stochastic noise.

Conversely, the multi-stage models required higher parameter counts to approach their respective (average) performance asymptotes. Because the *Stacked* and *Sidekick* models commit to a rigid, clean-data prior during Stage 1, a substantial portion of their Stage 2 parameter capacity is consumed by the necessity to overcome this structural bias. Consequently, while the performance of the *Stacked* model continues to increase with higher complexities, its trendline approaches an asymptote well below the peak of the *Monolith*. The *Sidekick* model hits a rigid ceiling near 150 parameters, as its latent-matching objective function artificially restricts the exploration of the wider parameter space.

Ultimately, complexity analysis identifies the phenomenon which we call the "Curse of Asymmetry". In asymmetric tasks, such as sequence denoising, modular and pre-trained QAEs require deeper circuits to achieve stability and more parameters to reach their performance ceilings. Unconstrained, end-to-end optimization not only discovers superior denoising manifolds but does so with vastly greater parameter efficiency. This aspect will be discussed in more detail in the following sections.

Table 2: Performance-to-effort analysis for the **best** QAEs training configurations, each configuration includes model instances of 6 qubits, 4 layers, and 4/2 latent-trash split.

QAE Architecture	S1	S2	Total Epochs	Δ_{test} (%)	Time (s)
Monolith	000	300	300*	67.15 ± 3.45	299
Sidekick (frozen)	500	500	1000	47.81 ± 10.19	657
Sidekick (retrained)	200	300	500	33.79 ± 21.66	372
Mirror	500	100	600**	40.01 ± 7.68	295
Stacked (frozen)	600	500	1100	49.99 ± 6.50	692
Stacked (retrained)	400	300	700	64.45 ± 4.31	525

* *S1 no model pretraining - single stage training.*

** *S2 included only model application to 100 noisy windows for evaluation.*

5.2 Denoising performance of all QAE architectures

To arrive at the performance-to-effort profiles of QAE architectural types within the stability zone and with sufficient complexity to rich the peak of their performance, we limited our analysis to models of 6 qubits, 4 layers, and 4/2 latent-trash split. Table 2 reports all such models and their *best* performing configuration (in terms of Stage 1 and Stage 2 epochs), their denoising performance in tests ($\Delta_{test}(\%)$), and the average "wall" time taken to optimize these models (Time (s)). The table takes into account all model architectures and their variants. It is also important to note that for each individually configured model, several (7) differently initialized instances were trained and tested. These final performance results identify the most important trends for the analyzed models.

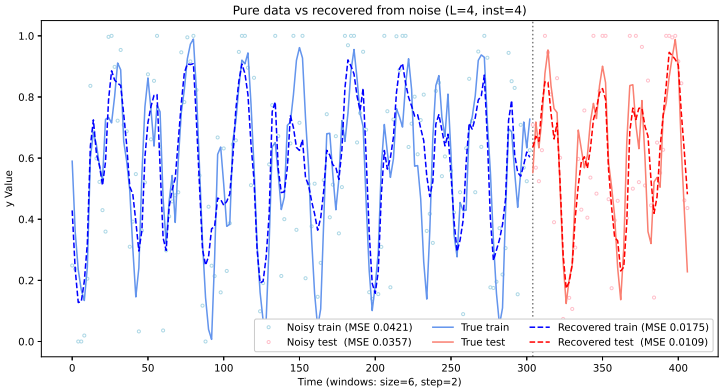
Winners and Losers. In a single pass of 300 epochs, at 67.15% (±3.45%) denoising performance (with the minimum variance), the *Monolith* QAE is the leader of the raw noise removal (Table 2). However, *Stacked (retrained)* is the close second, achieving 64.45% (±4.31%) denoising with an overall requirement of 700 epochs. The worst performing models are *Sidekick (retrained)* with its best denoising configuration of disappointing 33.79% (±21.66%) noise elimination (over 1000 epochs), and the *Mirror* with 40.01% (±7.68%, over 295 epochs).

It is important to recall that considering all structurally different architectures and all their training regimes, *Stacked (retrained)* achieved the highest average denoising performance of 61.73% and *Monolith* was second with 53.70% (Table 1).

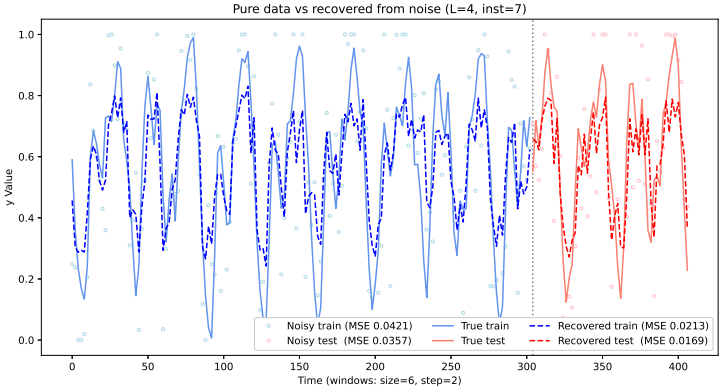
It is worth noting that among the best performing models, the *Monolith* training required the least wall time (299s), while both *frozen* models (*Sidekick* and *Stacked*) required the longest time (657s and 692s, respectively).

The visual assessment of the representative model instances of different QAE architectures and their fit to time series data (Figure 10) reveals that *frozen Stacked* and *Sidekick* demonstrated a similar level of denoising performance in training (from $MSE \approx 0.042$ to $MSE \approx 0.022$) and testing (from $MSE \approx 0.035$ to $MSE \approx 0.016$), *Monolith* surpassed them in training (from $MSE \approx 0.042$ to $MSE \approx 0.018$) and testing (from $MSE \approx 0.035$ to $MSE \approx 0.011$).

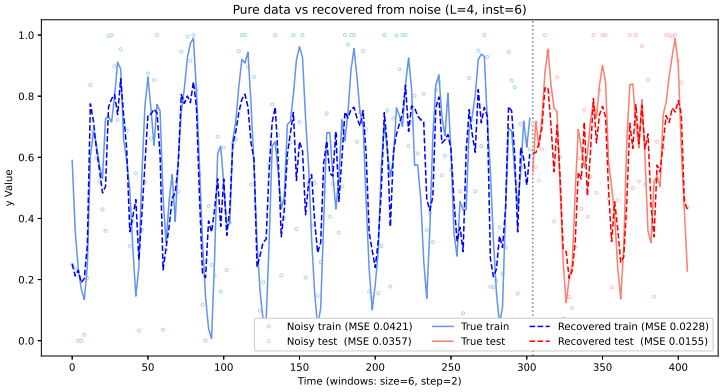
The Proxy-Metric Disconnect. The most critical revelation is the failure of the Sidekick's training objective, esp. when compared with other models (Table 2).



(a) Monolith fit to noisy time series (6q 4l 2t 4L inst4)



(b) Stacked fit to noisy time series (6q 4l 2t 4L inst7)



(c) Sidekick fit to noisy time series (6q 4l 2t 4L inst6)

Figure 10: Fitting QAEs *frozen* architectures to noisy time series (flattened)

Stacked (frozen) and *Sidekick (frozen)* exhibit similar denoising performance ($49.99 \pm 6.50\,\%$ and $47.81 \pm 10.19\,\%$, respectively). However, on retraining the performance of *Stacked (retrained)* and *Sidekick (retrained)* diverges significantly ($64.45 \pm 4.31\,\%$ and $33.79 \pm 21.66\,\%$, respectively). The spread of *Sidekick (retrained)* results also increased dramatically.

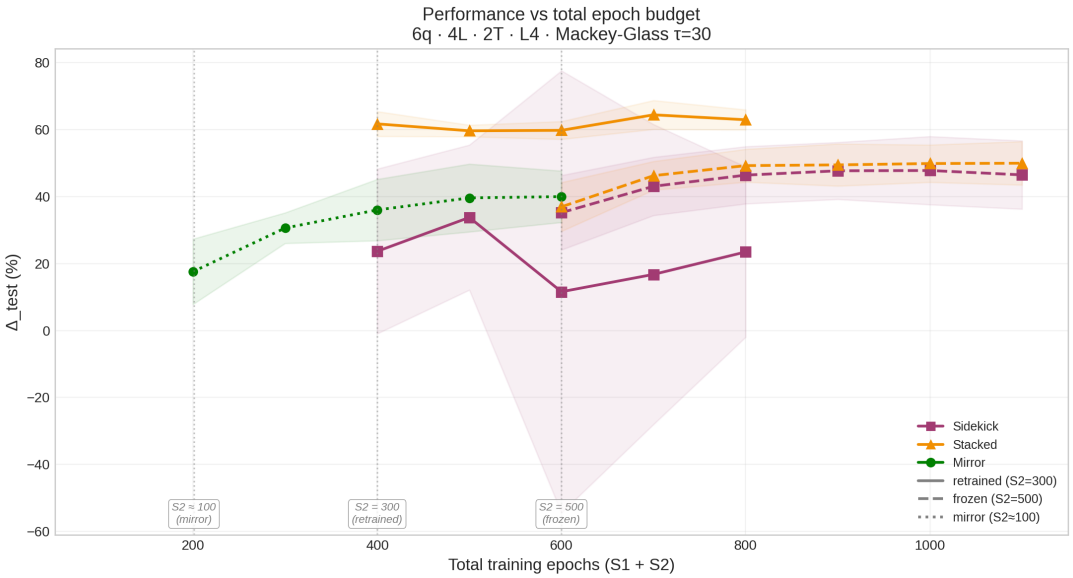


Figure 11: Impact of pretraining time vs denoising test performance (MG $\tau = 30$, models 6q 4l 2t L4). Here, the curve's starting point denotes the model performance achieved at the minimum pretraining and a fixed amount of Stage 2 training. The curve itself indicates performance with the increasing amount of Stage 1 pretraining.

As the *Stacked* models Stage 2 training is based directly on the final output reconstruction, their retraining success indicates that their Stage 1 pretraining brings their weights close to the optimal values. At the same time, the *Sidekick* models converge in Stage 2 by minimizing the SWAP-test distance in the latent space. The low denoising performance and large spread of results indicate that latent-space matching demands an exceptionally rigid and refined optimization target. Such an indirect approach to model optimization does not guarantee quality of the sequence reconstruction.

Therefore, convergence of the latent SWAP test is a flawed proxy for the denoising success.

Early over-specialization. A central challenge in the development of multi-stage Quantum Autoencoders (QAEs) is balancing the fidelity of the initial compression against the model's ultimate capacity for noise adaptation. Figure 11 (the curves denote only the pretraining epochs) maps the total epoch budget (S1+S2) to the percentage of noise removed in testing. The figure identifies an early saturation point within the relationship between pretraining duration and denoising efficacy – while extended Stage 1 training traditionally aims to sharpen the inverted decoder's representation of clean data, our results demonstrate that early over-specialization of QAE components exhausts its architecture learning capacity.

All multi-stage architectures, with the exception of *Sidekick (retrained)*, exhibit a very rapid performance saturation as pretraining increases. The *Stacked (retrained)* model gains very little from pretraining. In fact, with zero pretraining, *Stacked (retrained)* is equivalent to the superior *Monolith*. Conversely, when optimization is performed only in the pretraining phase, the *Stacked (retrained)* model is equivalent to the inferior *Mirror* model. This suggests that the 'clean' latent manifold defined in Stage 1 can become too rigid to accommodate

the stochastic nature of noisy inputs, effectively trapping the model in a local minimum that prioritizes mathematical symmetry over practical denoising robustness.

The behavior of the *Sidekick (retrained)* model presents a nuanced case of structural vs. optimization saturation. While the parameter-count analysis indicates representational saturation at approximately 150 parameters (Figure 9), the larger model continues to exhibit performance gains and reduced variance with extended pretraining (Figure 11). This suggests that for latent-matching architectures, the primary bottleneck is not the total expressive capacity of the circuit, but the stability of the optimization landscape. Extended pretraining likely provides a more robust 'initialization anchor' that partially mitigates the stochastic instability inherent in retraining dual-stage latent objectives.

Counter-intuitively, the more inflexible *Sidekick (frozen)* and *Stacked (frozen)* architectures continue to show marginal gains; because their encoders are anchored to a fixed latent target, extended pretraining serves to 'sharpen' that target, providing a more stable (if limited) objective for the Stage 2 optimization. It is also worth noting that structurally and performance-wise, *Stacked (frozen)* seems a natural extension of the *Mirror* architecture (Figure 11), which highlights the dependency of both model types on Stage 1 pretraining.

The Generalization Dilemma. In evaluating the denoising performance across all QAE variants (see Table 1), we observed a systemic baseline shift where the test-set noise removal (Δ_{test}) consistently exceeded the training-set noise removal (Δ_{train}) (also note the denoising distribution above the diagonal in Figure 7).

As this positive generalization gap occurred uniformly across all configurations, including unstable architectures with huge performance variance, it must be attributed to temporal data partitioning rather than universal model robustness.

The Mackey-Glass system at $\tau = 30$ is highly chaotic, characterized by distinct temporal regimes of varying localized complexity. To strictly prevent temporal data leakage, our training and testing datasets were partitioned chronologically. Consequently, the test partition must have captured the temporal features of the chaotic behavior with simpler local dynamics or lower variance than the training partition. This presented a universally 'easier' baseline for reconstruction. While this dataset asymmetry elevates the absolute test scores across the board, it acts as a uniform scalar; therefore, the relative performance hierarchies and structural stability comparisons between the *Monolith*, *Mirror*, *Stacked*, and *Sidekick* architectures remain rigorously valid.

These performance variations suggest that the underlying structural symmetry of the QAE governs its capacity for noise adaptation, a relationship explored in the following summary.

6 Summary and Conclusions

In this work, we systematically evaluated the efficacy of modular, multi-stage training paradigms versus end-to-end optimization for Quantum Autoencoders (QAEs) applied to the asymmetric task of chaotic time series denoising.

Our study of various Quantum Autoencoder configurations reveals that denoising efficacy is not merely a function of circuit depth, but of **Optimization Symmetry**. We conclude that:

- **Co-evolutionary Asymmetry (Monolith):** By bypassing Stage 1 pretraining, the Monolith avoids the **Curse of Asymmetry**. Its independent encoder and decoder co-evolve within a unified optimization manifold, allowing the model to achieve the highest

parameter efficiency. This asymmetry-induced synchronization enables the architecture to discover noise-resilient manifolds that rigid, symmetric priors cannot access.

- **Rigid Functional Asymmetry (Frozen Modular Design):** The two-stage frozen models, *Stacked (frozen)* and *Sidekick (frozen)*, are tethered to a "clean-data prior." Their performance acts as a direct, incremental extension of the single stage *Mirror* baseline, but they suffer from **Early Over-specialization**, which exhausts the model's limited representational budget on an unachievable mapping.

- **Asymmetric Handshake Instability (Retrained Modular Design):** This architecture, as featured in *Sidekick (retrained)*, exemplifies a **Proxy-Metric Disconnect**. By relying on the fidelity measure (using the SWAP-test) to maintain a latent-space handshake within the Hilbert space while simultaneously adapting to noise, the model lacks a stable optimization anchor. This is in contrast to *Stacked (retrained)*, which uses a simple pipeline of state evolution and relies on the outcome-based cost, and consequently achieving excellent denoising performance.

- **The Inversion Fallacy (The Mirror Lesson):** The comparatively inferior denoising performance of the *Mirror* QAE (as relative to the other QAE architectures) provides evidence against the imposition of strict structural symmetry in inherently asymmetric tasks. Owing to the architectural constraint that the encoder is defined as the adjoint of the decoder ($E = D^\dagger$), the model lacks independently learnable, noise-specific parameters capable of compensating for the statistical mismatch between noisy inputs and clean targets. This results in a "Symmetry Trap", i.e.

  - **Parameter Exhaustion.** The circuit's representational budget is entirely consumed by the requirement to invert a clean-data manifold.
  - **Functional Blindness.** Lacking facilities to deal with noise, *Mirror* treats stochastic variance as a clean signal, leading to structural mediocrity and poor results.
  - **Path-Dependency Bottleneck.** The *Mirror*'s failure shows that mathematical elegance ($E = D^\dagger$) is an architectural liability, as it prohibits the "asymmetry-induced synchronization" required to navigate chaotic manifolds.

These findings challenge the prevailing quantum machine learning heuristic, which suggests that deep variational circuits must be modularized and pre-trained to ensure effective convergence. While modularity aims for structural elegance, our results suggest that **Co-evolutionary Asymmetry** is the more robust path for NISQ-era denoising. The *Monolith's* ability to let its asymmetric components reach their own equilibrium leads to a stable and efficient model – without pretraining and handshaking with clean-data prior.

A final note of caution is warranted at this juncture. As *Monolith* style QAE architectures feature a large number of trainable parameters, they may be more susceptible to the emergence of *barren plateaus* during training. Should this occur, our findings suggest that a compromise – streamlined modular QAE design with pretraining as a warm-start, as seen in the *Stacked (retrained)* configuration – could mitigate the onset of barren plateaus while maintaining denoising performance comparable to that of the *Monolith*.

Our future investigations will focus on expanding this architectural comparison to varying degrees of system chaos to determine **Curse of Asymmetry** scaling with sequence entropy. The transition from quantum simulation to quantum hardware will also be critical.

Acknowledgement

References

[1] Asaoka, H., Kudo, K.: Quantum autoencoders for image classification. Quantum Machine Intelligence **7**(2), 71 (Dec 2025)

[2] Berahmand, K., Daneshfar, F., Salehi, E.S., Li, Y., Xu, Y.: Autoencoders and their applications in machine learning: A survey. Artificial Intelligence Review **57**(2), 28 (Feb 2024)

[3] Blázquez-García, A., Conde, A., Mori, U., Lozano, J.A.: A Review on Outlier/Anomaly Detection in Time Series Data. ACM Computing Surveys (CSUR) **54**(3), 1–33 (2021)

[4] Box, G.E.P., Jenkins, G.M., Reinsel, G.C., Ljung, G.M.: Time Series Analysis: Forecasting and Control. Wiley, 5 edn. (2015)

[5] Bravo-Prieto, C.: Quantum autoencoders with enhanced data encoding. Machine Learning: Science and Technology **2** (May 2021)

[6] Chen, S., Guo, W.: Auto-Encoders in Deep Learning—A Review with New Perspectives. Mathematics **11**(8), 1777 (Jan 2023)

[7] Chiarot, G., Silvestri, C.: Time series compression: A survey. ACM Computing Surveys **55**(10), 1–32 (Oct 2023)

[8] Cunningham, J., Zhuang, J.: Investigating and mitigating barren plateaus in variational quantum circuits: A survey. Quantum Information Processing **24**(2) (Jan 2025). https://doi.org/10.1007/s11128-025-04665-1

[9] Cybulski, J.L., Nguyen, T.: Impact of barren plateaus countermeasures on the quantum neural network capacity to learn. Quantum Information Processing **22**(12), 442 (Dec 2023)

[10] Cybulski, J.L., Zając, S.: Design Considerations for Denoising Quantum Time Series Autoencoder. In: Franco, L., de Mulatier, C., Paszynski, M., Krzhizhanovskaya, V.V., Dongarra, J.J., Sloot, P.M.A. (eds.) Computational Science – ICCS 2024. pp. 252–267. Springer Nature Switzerland, Cham (2024)

[11] Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. The MIT Press (Nov 2016)

[12] Grant, E., Wossnig, L., Ostaszewski, M., Benedetti, M.: An initialization strategy for addressing barren plateaus in parametrized quantum circuits. Quantum **3**, 214 (Dec 2019)

[13] Jia, X., Xun, P., Peng, W., Zhao, B., Li, H., Shen, C.: Deep anomaly detection for time series: A survey. Computer Science Review **58**, 100787 (Nov 2025)

[14] Kalman, R.E.: A new approach to linear filtering and prediction problems. Journal of Basic Engineering **82**(1), 35–45 (1960)

[15] Kaur, A., Dong, G.: A Complete Review on Image Denoising Techniques for Medical Images. Neural Processing Letters **55**(6), 7807–7850 (Dec 2023)

[16] Khoshaman, A., Vinci, W., Denis, B., Andriyash, E., Sadeghi, H., Amin, M.H.: Quantum variational autoencoder. Quantum Science and Technology **4**(1), 014001 (Sep 2018)

[17] Kortemme, T.: De novo protein design—From new structures to programmable functions. Cell **187**(3), 526–544 (Feb 2024). https://doi.org/10.1016/j.cell.2023.12.028

[18] Lapedes, A., Farber, R.: Nonlinear signal processing using neural networks: Prediction and system modelling (LA-UR-87-2662; CONF-8706130-4) (Jun 1987)

[19] Li, P., Pei, Y., Li, J.: A comprehensive survey on design and application of autoencoder in deep learning. Applied Soft Computing **138**, 110176 (May 2023)

[20] Liu, C.J., Liu, H.T., Bian, C., Chen, X.D., Yang, S.H., Wang, X.F.: Investigation of Time-series Prediction for Turbine Machinery Condition Monitoring. IOP Conference Series: Materials Science and Engineering **1081**(1), 012022 (Feb 2021)

[21] Liu, T., Quan, Y., Su, Y., Guo, Y., Liu, S., Ji, H., Hao, Q., Gao, Y., Liu, Y., Wang, Y.: Astronomical image denoising by self-supervised deep learning and restoration processes. Nature Astronomy **9**(4), 608–615 (2025)

[22] Lu, L., Yin, K.L., de Lamare, R.C., Zheng, Z., Yu, Y., Yang, X., Chen, B.: A survey on active noise control in the past decade—Part I: Linear systems. Signal Processing **183**, 108039 (Jun 2021)

[23] Lyu, C., Xu, X., Yung, M.H., Bayat, A.: Symmetry enhanced variational quantum spin eigensolver. Quantum **7**, 899 (Jan 2023)

[24] Mackey, M.C., Glass, L.: Oscillation and Chaos in Physiological Control Systems. Science **197**(4300), 287–289 (Jul 1977). https://doi.org/10.1126/science.267326

[25] Mangini, S., Marruzzo, A., Piantanida, M., Gerace, D., Bajoni, D., Macchiavello, C.: Quantum neural network autoencoder and classifier applied to an industrial case study. Quantum Machine Intelligence **4**(2), 13 (Jun 2022)

[26] Mari, A., Bromley, T.R., Izaac, J., Schuld, M., Killoran, N.: Transfer learning in hybrid classical-quantum neural networks. Quantum **4**, 340 (Oct 2020)

[27] Mastriani, M.: Quantum spectral analysis: Frequency in time, with applications to signal and image processing (Feb 2021)

[28] McArdle, S., Endo, S., Aspuru-Guzik, A., Benjamin, S.C., Yuan, X.: Quantum computational chemistry. Reviews of Modern Physics **92**(1), 015003 (Mar 2020)

[29] Meyer, J.J., Mularski, M., Gil-Fuster, E., Mele, A.A., Arzani, F., Wilms, A., Eisert, J.: Exploiting Symmetry in Variational Quantum Machine Learning. PRX Quantum **4**(1), 010328 (Mar 2023)

[30] Mok, W.K., Zhang, H., Haug, T., Luo, X., Lo, G.Q., Cai, H., Kim, M.S., Liu, A.Q., Kwek, L.C.: Rigorous noise reduction with quantum autoencoders (Aug 2023)

[31] Ngairangbam, V.S., Spannowsky, M., Takeuchi, M.: Anomaly detection in high-energy physics using a quantum autoencoder. Physical Review D **105**(9), 095004 (May 2022)

[32] Nielsen, M., Chuang, I.: Quantum Computation and Quantum Information: 10th Anniversary Ed. Cambridge University, New York (2010)

[33] Nishimura, H.: A Survey: SWAP Test and Its Applications to Quantum Complexity Theory. In: Minato, S.i., Uno, T., Yasuda, N., Horiyama, T., Kawarabayashi, K.i., Yamashita, S., Ono, H. (eds.) Algorithmic Foundations for Social Advancement, pp. 243–261. Springer Nature Singapore, Singapore (2025)

[34] Pfeiffer, J., Ruder, S., Vulić, I., Ponti, E.M.: Modular Deep Learning (Jan 2024). https://doi.org/10.48550/arXiv.2302.11529

[35] Qiao, J., Gao, W., Jin, J., Wang, D., Guo, X., Manavalan, B., Wei, L.: Molecular pretraining models towards molecular property prediction. Science China Information Sciences **68**(7), 170104 (2025)

[36] Rivas, P., Zhao, L., Orduz, J.: Hybrid Quantum Variational Autoencoders for Representation Learning. In: 2021 International Conference on Computational Science and Computational Intelligence (CSCI). pp. 52–57. IEEE, Las Vegas, NV, USA (Dec 2021)

[37] Romero, J., Olson, J.P., Aspuru-Guzik, A.: Quantum autoencoders for efficient compression of quantum data. Quantum Science and Technology **2**(4), 045001 (2017)

[38] Rudin, L.I., Osher, S., Fatemi, E.: Nonlinear total variation based noise removal algorithms. Physica D: Nonlinear Phenomena **60**(1–4), 259–268 (1992)

[39] Sakhnenko, A., O'Meara, C., Ghosh, K.J., Mendl, C.B., Cortiana, G., Bernabé-Moreno, J.: Hybrid classical-quantum autoencoder for anomaly detection. Quantum Machine Intelligence **4**(2), 27 (2022)

[40] Sezer, O.B., Gudelek, M.U., Ozbayoglu, A.M.: Financial Time Series Forecasting with Deep Learning : A Systematic Literature Review: 2005-2019 (Nov 2019)

[41] Skolik, A., McClean, J.R., Mohseni, M., van der Smagt, P., Leib, M.: Layerwise learning for quantum neural networks. Quantum Machine Intelligence **3**(1), 5 (Jan 2021)

[42] Sörnmo, L., Laguna, P.: Bioelectrical Signal Processing in Cardiac and Neurological Applications. Elsevier (2005)

[43] Tripathi, P.M., Kumar, A., Komaragiri, R., Kumar, M.: A Review on Computational Methods for Denoising and Detecting ECG Signals to Detect Cardiovascular Diseases. Archives of Computational Methods in Engineering **29**(3), 1875–1914 (May 2022)

[44] Wiener, N.: Extrapolation, Interpolation, and Smoothing of Stationary Time Series. MIT Press (1949)

[45] Wu, J., Fu, H., Zhu, M., Zhang, H., Xie, W., Li, X.Y.: Quantum circuit autoencoder. Physical Review A **109**(3), 032623 (Mar 2024). https://doi.org/10.1103/PhysRevA.109.032623