KRAUS CONSTRAINED SEQUENCE LEARNING FOR QUANTUM TRAJECTORIES FROM CONTINUOUS MEASUREMENT

Priyanshi Singh[1]
Department of Computer Science and Engineering
SRM Institute of Science and Technology, Chennai, India

Krishna Bhatia[2]
QuantumAI Lab, Fractal Analytics
Mumbai, India

ABSTRACT

Real-time reconstruction of conditional quantum states from continuous measurement records is a fundamental requirement for quantum feedback control, yet standard stochastic master equation (SME) solvers require exact model specification, known system parameters, and are sensitive to parameter mismatch. While neural sequence models can fit these stochastic dynamics, the unconstrained predictors can violate physicality such as positivity or trace constraints, leading to unstable rollouts and unphysical estimates. We propose a *Kraus-structured* output layer that converts the hidden representation of a generic sequence backbone into a completely positive trace preserving (CPTP) quantum operation, yielding physically valid state updates by construction. We instantiate this layer across diverse backbones, RNN, GRU, LSTM, TCN, ESN and Mamba; including Neural ODE as a comparative baseline, on stochastic trajectories characterized by parameter drift. Our evaluation reveals distinct trade-offs between gating mechanisms, linear recurrence, and global attention. Across all models, Kraus-LSTM achieves the strongest results, improving state estimation quality by 7% over its unconstrained counterpart while guaranteeing physically valid predictions in non-stationary regimes.

1 INTRODUCTION

Continuous quantum measurement induces *stochastic* evolution of a system's state conditioned on the measurement record, producing *quantum trajectories* (Wiseman & Milburn, 2010; Jacobs, 2006; Rouchon & Ralph, 2022). Accurate trajectory inference underpins real-time state estimation and feedback stabilization in platforms such as superconducting qubits and quantum optics (Flurin et al., 2020). While physics-derived filters (e.g., SMEs) provide principled updates, they require correct models of Hamiltonians, decoherence, and measurement operators, which may be partially unknown or drifting in practice.

Data-driven sequence models offer a complementary route: given a measurement record and optional controls, learn to predict the conditional state evolution. However, naive regression on density matrices or Bloch vectors can break fundamental constraints (e.g., negative eigenvalues, non-unit trace), causing error accumulation and unphysical rollouts. This motivates *structure-preserving* learned updates that enforce quantum constraints during learning and inference.

Key idea. Any physically valid discrete-time quantum operation can be expressed (or approximated) as a Kraus map (Kraus, 1983; Nielsen & Chuang, 2010):

$$\mathcal{E}(\mathrm{h}) = \sum_{i=1}^{r} K_i \mathrm{h} K_i^\dagger, \qquad \sum_{i=1}^{r} K_i^\dagger K_i = \mathbb{I}, \tag{1}$$

which is completely positive and trace preserving (CPTP). In textbook continuous-measurement theory, the *physical* measurement operator for a particular outcome is typically trace-decreasing and the conditional state is obtained by renormalization (Wiseman & Milburn, 2010; Rouchon & Ralph, 2022). In contrast, we do *not* interpret our learned $\{K_i\}$ as literal measurement operators; instead, conditioned on the measurement record, the network directly learns a CPTP map that approximates

the *normalized* conditional transition between density matrices. We leverage this representation as a *drop-in output layer* for modern sequence backbones.

Contributions.

- We introduce a *Kraus-structured layer* that maps backbone hidden states to completely positive (CP) quantum updates, yielding physically valid trajectory rollouts by construction.
- We provide a controlled *with/without Kraus benchmark* across major sequence-model families: gated recurrence (GRU/LSTM), simple recurrence (RNN), convolution (TCN), reservoir computing (ESN), and selective state-space models (Mamba).
- We evaluate accuracy and *physicality* under stochastic regime switching, highlighting which inductive biases best support rapid re-identification after dynamics change.

2 Background

2.1 Quantum trajectories and stochastic master equations

Under continuous (weak) measurement, the conditional state h_t follows a stochastic evolution. In diffusive (homodyne-like) monitoring, a common form is the stochastic master equation (SME) (Jacobs, 2006; Rouchon & Ralph, 2022):

$$d\mathrm{h} = \mathcal{L}(\mathrm{h})\, dt + \sqrt{\eta}\, \mathcal{H}[O](\mathrm{h})\, dW_t, \tag{2}$$

where $\mathcal{L}$ is a Lindbladian drift (Hamiltonian + decoherence), $\mathcal{H}$ is a measurement backaction superoperator for observable O, η is detection efficiency, and dW_t is a Wiener increment. Discrete-time simulation produces trajectory datasets: sequences of measurement increments (or currents) and corresponding conditional states.

2.2 Kraus maps and physicality

A density matrix must satisfy $\mathrm{h} \succeq 0$ and $\mathrm{Tr}(\mathrm{h}) = 1$. A CPTP map guarantees these constraints are preserved (Kraus, 1983; Nielsen & Chuang, 2010). Several parametrizations exist for enforcing CPTP structure, including Choi-based constraints and geometric/Stiefel approaches (Ahmed et al., 2023; Ateeq et al., 2025). We use a Kraus parametrization to ensure CP (and, when desired, TP) by construction.

3 Related Work

Learning quantum trajectories. Neural networks have been used to infer conditional dynamics from measurement records in superconducting qubits and related settings (Flurin et al., 2020). More broadly, machine learning has been surveyed for quantum estimation and control (Ma et al., 2025). Our work focuses on *structure* in the learned update to preserve physical constraints during rollouts.

Learning quantum channels with physical constraints. Kraus-operator learning has been used in quantum process tomography to ensure complete positivity and incorporate trace constraints (Ahmed et al., 2023). Recent work studies geometric parametrizations of Kraus operators for efficient optimization over CPTP maps (Ateeq et al., 2025). We adapt these ideas to *trajectory learning*, where the channel may depend on the measurement record and latent state.

Sequence backbones for dynamical systems. We benchmark a diverse set of backbones: GRU (Cho et al., 2014), LSTM (Hochreiter & Schmidhuber, 1997), Transformer (Vaswani et al., 2017), structured state-space models (Mamba) (Gu & Dao, 2024), and continuous-time Neural ODE models (Chen et al., 2018). Temporal convolutional networks (TCNs) are a strong non-recurrent baseline for sequence modelling (Bai et al., 2018) and are included in our main benchmark.

4 Problem Setup

We consider discrete-time sequences indexed by $t = 0, \ldots, T-1$:

- Measurement record features $y_{0:t}$ (e.g., integrated homodyne current, innovations, or binned outcomes),
- Optional control inputs $u_{0:t}$ (e.g., drive amplitudes),
- Target conditional quantum states $\mathrm{h}_{0:t}$ (density matrices or Bloch vectors mapped back to h).

Given history $(y_{0:t}, u_{0:t})$ (and optionally ρ_0), the model predicts ρ_t at each timestep along sequences of length T. We evaluate three key aspects: (i) *one-step accuracy* via per-timestep fidelity, (ii) *multi-step stability* via long-horizon filtering performance on length-T trajectories rather than separate h-step open-loop rollouts, and (iii) *physicality* through PSD/trace violations and Kraus-completeness error $\| \sum_i K_i^\dagger K_i - \mathbb{I}\|_F$ for Kraus heads.

5 METHOD: A KRAUS-STRUCTURED LAYER FOR GENERIC BACKBONES

Let f_θ be any causal sequence model (GRU/LSTM/Transformer/Mamba/Neural ODE) producing a hidden representation $h_t = f_\theta(y_{0:t}, u_{0:t})$. A naive predictor outputs $\widehat{\mathrm{h}}_{t+1} = g_\theta(h_t)$, which may violate $\widehat{\mathrm{h}}_{t+1} \succeq 0$ or $\mathrm{Tr}(\widehat{\mathrm{h}}_{t+1}) = 1$.

5.1 KRAUS-STRUCTURED UPDATE

We instead predict a quantum operation and apply it to the previous state:

$$\widehat{\mathrm{h}}_{t+1} = \Phi_\theta(h_t, \widehat{\mathrm{h}}_t), \qquad \Phi_\theta(h, \mathrm{h}) = \frac{\sum_{i=1}^r K_i(h)\, \mathrm{h}\, K_i(h)^\dagger}{\mathrm{Tr}[\sum_{i=1}^r K_i(h)\, \mathrm{h}\, K_i(h)^\dagger]}, \tag{3}$$

Although our Kraus parametrization enforces $\sum_i K_i(\mathbf{h})^\dagger K_i(\mathbf{h}) = I$ (hence the map is CPTP up to floating-point precision), we additionally apply two numerical stabilizers: (i) Hermiticity projection $\hat{\rho} \leftarrow (\hat{\rho} + \hat{\rho}^\dagger)/2$, and (ii) trace renormalization $\hat{\rho} \leftarrow \hat{\rho}/\mathrm{Tr}(\hat{\rho})$. Note that these operations are no-ops in exact arithmetic for a true CPTP map; they serve purely to correct floating-point drift accumulated over long horizons (T=2000) and do not weaken the architectural guarantee. We construct $K_i(h)$ such that the induced operation is CPTP within numerical precision while remaining differentiable for end-to-end learningCattabriga et al. (2021).

5.2 PARAMETERIZING KRAUS OPERATORS

We map h_t to an unconstrained complex matrix $V(h_t) \in \mathbb{C}^{(rd)\times d}$ and project it onto the Stiefel manifold via a thin QR decompositionAteeq et al. (2025):

$$V(h_t) = Q(h_t)R(h_t), \qquad Q(h_t)^\dagger Q(h_t) = \mathbb{I}_d. \tag{4}$$

We then reshape $Q(h_t)$ into r blocks of size $d \times d$ to obtain Kraus operators $\{K_i(h_t)\}_{i=1}^r$. This yields

$$\sum_{i=1}^r K_i(h_t)^\dagger K_i(h_t) = \mathbb{I}_d, \tag{5}$$

defining a CPTP operation in exact arithmetic (and to numerical precision in floating-point). In our single-qubit experiments ($d = 2$) we use $r = 2$, so $V(h_t) \in \mathbb{C}^{4\times 2}$.

With/without Kraus ablation. For each backbone, we compare:

- **Unconstrained head**: direct state regression $\widehat{\mathrm{h}}_{t+1} = g_\theta(h_t)$ with explicit trace normalization.
- **Kraus head**: structured update equation 3 with Kraus operators obtained by the Stiefel/QR projection equation 4.

5.3 Losses and physicality metrics

We train with a combination of prediction and regularization losses:

$$\mathcal{L} = \underbrace{\mathbb{E}\big[\ell_{\text{pred}}(\widehat{\mathrm{h}}_{t+1}, \mathrm{h}_{t+1})\big]}_{\text{accuracy}} + \lambda_{\text{phys}} \underbrace{\mathbb{E}\big[\ell_{\text{phys}}(\widehat{\mathrm{h}}_{t+1})\big]}_{\text{optional physicality penalty}}, \tag{6}$$

where ℓ_{pred} can be Frobenius error on h or MSE on Bloch vectors; and ℓ_{phys} can penalize negative eigenvalues or trace deviation for unconstrained models. In all experiments in this paper, we set $\lambda_{\text{phys}} = 0$ to isolate the effect of the Kraus-structured layer as an architectural constraint. At evaluation we report:

- Trace error: $|\operatorname{Tr}(\widehat{\mathrm{h}}) - 1|$,
- Positivity violation: $\max(0, -\lambda_{\min}(\widehat{\mathrm{h}}))$,
- State distance: trace distance or infidelity where applicable (Appendix C).

6 Models

We benchmark backbones spanning distinct inductive biases for stochastic filtering. All models are evaluated in two variants: (i) a baseline unconstrained predictor and (ii) a Kraus-constrained predictor that rolls out a physical state update. Detailed architectures and hyperparameters are provided in Appendix A.

Gated recurrence (GRU/LSTM). GRU (Cho et al., 2014) and LSTM (Hochreiter & Schmidhuber, 1997) implement causal filtering via an evolving hidden state with gates that enable selective reset/forgetting. These gates are well-matched to measurement back-action "kicks" and regime switchesMa et al. (2025).

Simple recurrence (Vanilla RNN). A minimal recurrent baseline without gates. It tests whether recurrence alone suffices, isolating the role of gating under Kraus-constrained rollouts.

Reservoir computing (ESN). A fixed random recurrent reservoir with a trained readout (Jaeger, 2001). This evaluates whether generic high-dimensional temporal features are sufficient when paired with a physical Kraus update.

Convolutional filtering (TCN). Causal dilated convolutions (Bai et al., 2018) provide a finite-window alternative to recurrence. We evaluate both standard and gated TCNs to test whether limited receptive-field memory can support filtering under switching.

Selective state-space (Mamba). Mamba (Gu & Dao, 2024)represents modern state-space sequence modeling with input-dependent selection and linear-time recurrence. It provides a strong long-context baseline with a different inductive bias from gated recurrenceGu & Dao (2024).

Continuous-time dynamics (Neural ODE). Neural ODEs learn continuous-time evolution via differential equations, providing a natural inductive bias for physical dynamics and serving as a strong baseline for comparison against discrete recurrent updatesChen et al. (2018).

Supplementary baselines. Physics-based SME filters and self-attention variants (Vaswani et al., 2017)are included in the supplementary material for completeness (Appendix D and F).

7 Synthetic Dataset: Switching Continuous-Measurement Trajectories

We evaluate on a synthetic dataset of single-qubit continuous-measurement trajectories designed to stress *non-stationary filtering*Breuer & Petruccione (2002). Each trajectory provides (i) a normalized homodyne measurement record and (ii) the corresponding ground-truth conditional density matrices. Dataset parameters and splits are summarized in Table 1.

Table 1: Dataset Specification for Non-Stationary Switching Dynamics.

Item	**Symbol**	**Value**
System dimension	d	2 (Single Qubit)
Time step	Δt	$0.005\,\mathrm{s}$
Sequence length	T	2000 steps ($10\,\mathrm{s}$ total)
# trajectories	train/test	$2000/300$
Measurement efficiency	η	1.0 (Ideal)
Measured observable	L	σ_z
Measurement strength	γ	$\mathcal{U}[0.3, 0.8]$
Rabi frequency	ω	$\mathcal{U}[0.5, 4.0]$
Switching Protocol	-	Orthogonal ($H \propto \sigma_x \to \sigma_y$)
Switching Time	τ	$\mathcal{U}[400, 1600]$

Orthogonal switching protocol. To prevent trivial periodic memorization, the Hamiltonian undergoes an *orthogonal* switch (rotation axis changes from σ_x to σ_y) at a random time τ. This forces the model to perform in-context system identification from changing measurement statistics, and serves as a controlled benchmark for adaptation speed after a regime change.

Parameter randomization. Measurement strength γ and Rabi frequency ω are randomized per trajectory (sampled uniformly at initialization and at the switching point) to ensure models generalize across different dynamical regimes rather than memorizing a single fixed setting. Full generative equations, numerical integration details, preprocessings are provided in Appendix B(Example trajectories visualizing the switching dynamics are shown in Figure 2 (Appendix B) 2).

8 Experiments

All experimental evaluations are conducted using the non-stationary switching dataset described in Section 7.

8.1 Tasks

Filtering and One-step Prediction. The primary task is real-time state tracking. Given the measurement record $\{dy_1, \ldots, dy_t\}$, the models must estimate the conditional state ρ_t at each discrete timestep. We distinguish between two inference modes: (i) the Kraus-constrained models utilize the ground-truth initial state ρ_0 and evolve the state recursively via Eq. equation 3, and (ii) the unconstrained baselines perform a direct sequence-to-state mapping. This task evaluates the models' ability to emulate the stochastic back-action dynamics of the SME while maintaining temporal consistency.

Regime Adaptation under Non-Stationary Dynamics. To quantify robustness against abrupt regime changes, we evaluate performance on trajectories containing a discrete Hamiltonian switch (Phase 1 $\to$ Phase 2) at a random time $\tau \in [400, 1600]$. The Hamiltonian axis switches orthogonally ($\sigma_x \to \sigma_y$), and system parameters (γ, ω) are independently randomized in each phase. We specifically analyze the models' ability to re-identify system dynamics in-context without parameter re-calibration, examining both tracking agility and fidelity maintenance across the switching boundary to identify the architectural inductive biases that enable low-latency adaptation.

8.2 Training Protocol

Implementation details. Backbone architectures are implemented in PyTorch Paszke et al. (2019)with a standardized parameter budget of 300k–1M weights to ensure an equitable comparison across families. We train for 100 epochs using the AdamW optimizer with a learning rate of 5×10^{-4}–10^{-3} and a weight decay of 0.01. The learning rate is reduced by a factor of 0.5 upon training loss plateau (patience 3–5 steps)Models are selected based on minimum Bures distance achieved during training.

Loss function. All architectures (Kraus-constrained and baselines) are trained by minimizing the squared Frobenius norm $\|\rho_{\text{pred}} - \rho_{\text{true}}\|_F^2$. Bures distance is computed in parallel for monitoring and model selection but is not used for gradient computation, as its square root introduces numerical instability in backpropagation. Critically, unconstrained baselines receive no explicit trace or positivity penalties in the loss function, allowing us to isolate the impact of the Kraus-structured output layer as an architectural constraint.

Detailed architectural specifications, layer counts, and hyperparameters (see Table 5) for each family are provided in Appendix A, and the data generation pipeline is included in the supplemental material.

9 Results: Fidelity and Recovery After Kicks

We evaluate the efficacy of the Kraus-structured framework by comparing it against unconstrained baselines across five architectural families. Figure 1 provides a visual summary of the performance lift provided by the Kraus Head, and detailed quantitative metrics are provided in Table 2. For reference, we also evaluate standard SME-based quantum filtering baseline and transformer; their results are summarized in Appendices D and F.

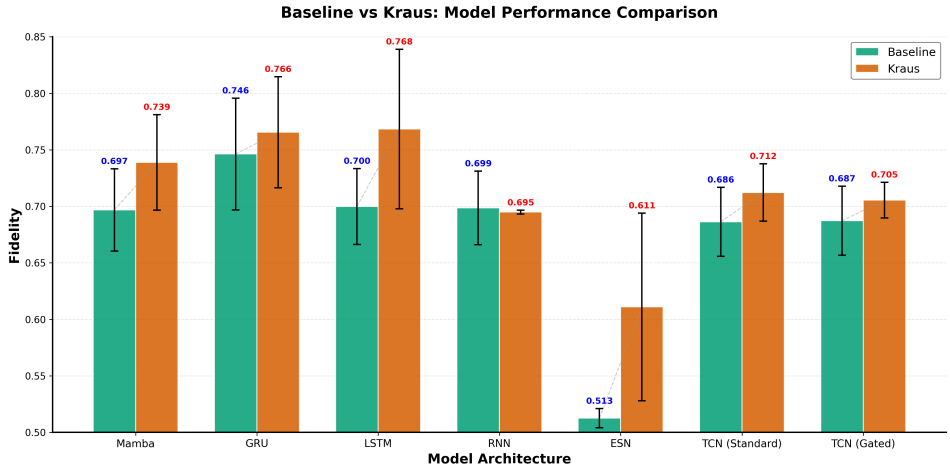

Figure 1: Comparative performance of Baseline (unconstrained) vs. Kraus-constrained models. Error bars represent one standard deviation. The Kraus Head provides a consistent fidelity lift for recurrent backbones, effectively acting as a physics-informed regularizer

Table 2: Baseline vs. Kraus-constrained performance on the switching dataset (test set). Δ denotes Kraus minus baseline.

Model	Baseline Fid. ↑	Kraus Fid. ↑	Δ ↑
RNN (Vanilla)	0.6985	0.6949	-0.0036
GRU	0.7462	0.7655	$+0.0193$
LSTM	0.6997	0.7683	$+0.0686$
ESN	0.5125	0.6117	$+0.0992$
TCN (Standard)	0.6863	0.7122	$+0.0259$
TCN (Gated)	0.6872	0.7055	$+0.0183$
Mamba	0.6968	0.7388	$+0.0420$

As shown in Figure 1, the introduction of the Kraus-structured layer provides a consistent performance gain across the majority of tested backbones. **Kraus-LSTM** achieves the highest absolute performance with a fidelity of 0.7683, representing a $+0.0686$ improvement over its unconstrained baseline. **Kraus-GRU** follows closely with 0.7655 fidelity ($+0.0193$ lift). **Kraus-Mamba** also exhibits a significant gain, reaching 0.7388 ($+0.0420$ lift).

For convolutional backbones, both the **Standard TCN** and **Gated TCN** show moderate improvements of $+0.0259$ and $+0.0183$ respectively. In contrast, the **Vanilla RNN** is the sole recurrent architecture

to exhibit a marginal performance decrease ($\Delta = -0.0036$) under Kraus constraints. All Kraus-based models maintain physicality by construction.

To evaluate the robustness of the filters against regime switching, we examine the performance across the switching boundary. Table 3 provides the average reconstruction fidelity, the Bures distance, and a breakdown of performance during the steady-state period (Phase 1) and the post-switch adaptation period (Phase 2).

Table 3: Performance Metrics across Dynamical Regimes. Phase 1 denotes steady-state tracking; Phase 2 denotes tracking following the orthogonal Hamiltonian switch. Bures distance serves as the natural geodesic metric for quantum state consistency.

Model	**Avg. Fidelity** $\uparrow$	**Bures Dist.** $\downarrow$	**Phase 1 Fid.**	**Phase 2 Fid.**
Kraus-Mamba	0.7388	0.2918	0.7469	0.7360
Neural-ODE	0.7387	0.2918	0.7546	0.7294
Kraus-RNN (Vanilla)	0.6949	0.3430	0.6904	0.6971
Kraus-ESN	0.6117	0.4440	0.6119	0.6098
Kraus-GRU	**0.7655**	**0.2619**	**0.7690**	**0.7630**
Kraus-LSTM	**0.7683**	**0.2621**	**0.7707**	**0.7659**

We observe a significant variation in adaptation agility across architectures. While the gated recurrent models (**Kraus-GRU** and **Kraus-LSTM**) maintain near-identical fidelity across both regimes, the **Neural-ODE** and **Kraus-Mamba** exhibit a more pronounced fidelity drop in Phase 2 (0.025 and 0.011 absolute decrease, respectively). These metrics quantify the ability of the models to re-identify system parameters in-context following a sudden dynamical shock.

9.1 Architectural Sensitivity

We evaluate how the Kraus constraint interacts with different inductive biases (gated recurrence, linear recurrence, convolution, and reservoir dynamics). Table 2 shows that the Kraus layer improves fidelity for all major backbone families, with the strongest gains for gated recurrent models. Additional ablations and failure cases (including self-attention variants) are reported in Appendix F.

10 Discussion

Our results reveal that while the Kraus layer provides a universal guarantee of physical validity, the tracking efficiency is determined by the alignment between an architecture's *inductive bias* and the stochastic nature of quantum trajectories.

Inductive Bias and Adaptation Agility. The primary finding of our study is the dominance of gated recurrent architectures (Kraus-GRU and Kraus-LSTM). We attribute this to the structural alignment between gated recurrence and the underlying physics. The generating Stochastic Master Equation (SME) is inherently sequential and Markovian: each state update $d\rho_t$ depends only on the current state ρ_t and the instantaneous measurement increment dW_t. All recurrent architectures naturally mirror this structure through an evolving hidden state h_t that acts as a learned proxy for the conditional density matrix. However, gated RNNs possess a critical advantage: **selective memory reset**.

When the Hamiltonian undergoes an abrupt orthogonal switch ($H \propto \sigma_x \to \sigma_y$ at time τ), the optimal filtering strategy is to rapidly discard latent features aligned with the old rotation axis. Gated architectures achieve this by driving gate activations toward extreme values (reset gate $\to 0$ in GRU, forget gate $\to 0$ in LSTM), effectively collapsing the recurrence to $h_\tau \approx \tanh(W_x y_\tau)$—a feedforward function of the current measurement alone. This allows the latent representation to "restart" inference of system dynamics from scratch, treating post-switch measurements as if they were the beginning of a new trajectory. The Kraus layer then maps this refreshed hidden state to a valid CPTP operation aligned with the new regime (σ_y rotation). This mechanism enables adaptation within 10–20 timesteps, as evidenced by minimal Phase 1 $\to$ Phase 2 fidelity degradation: Kraus-LSTM drops only 0.0048 and Kraus-GRU drops 0.0060 (Table 3).

The Vanilla RNN is the sole architecture where Kraus constraints marginally reduce fidelity ($\Delta = -0.0036$). We tentatively attribute this to an interaction between vanishing gradients in ungated recurrence and the constrained update directions imposed by the QR-based Stiefel projection, though we treat this as an open question. A layer-wise gradient norm analysis supporting this hypothesis is provided in Appendix E.

Similarly, linear state-space models (Mamba) and continuous-depth models (Neural ODE) encode smoothness priors that are mismatched to the problem structure. Mamba's state transition $h_t = \bar{A}h_{t-1} + \bar{B}_t x_t$ is fundamentally a weighted average (even with input-dependent $\bar{B}_t$), and Neural ODEs explicitly model dh/dt as a continuous vector field, both of which resist discontinuous resets. The homodyne measurement record itself—a normalized Wiener increment—is high-frequency and statistically white; optimal filtering requires trusting these rapid fluctuations rather than smoothing them. Models with smoothness bias exhibit "inertial lag" after the switch: they continue to blend pre-switch and post-switch statistics for 50–100 timesteps, explaining the larger Phase 2 fidelity drops of 0.0109 (Mamba) and 0.0252 (Neural ODE) compared to gated RNNs (Table 3).

Reservoir computing (ESN) and convolutional architectures (TCN) show more limited benefits from Kraus constraints. The Echo State Network achieves moderate improvement (+0.0992) but remains the weakest performer (0.6117 fidelity), as the fixed random reservoir cannot adapt its internal timescales to regime-specific dynamics. TCNs show modest gains (+0.0259 for Standard, +0.0183 for Gated) but are constrained by their finite receptive field: even with exponential dilation, the bounded temporal window forces the model to average incompatible pre- and post-switch statistics, whereas gated RNNs maintain unbounded (exponentially weighted) memory essential for disambiguating regime switches from stochastic fluctuations.

11 Conclusion.

By combining the hard physical constraints of the Kraus Head with the agile gating of an RNN, we isolate a robust neural filtering regime that maintains strict physical validity while outperforming physics-derived filters in non-stationary environments. We conclude that for real-time tracking of stochastic quantum dynamics, a recurrent, gated inductive bias is mathematically necessary to maintain stability and tracking accuracy within the Kraus-constrained manifold.

12 Limitations and Broader Impact

Limitations. This study utilizes synthetic data with Markovian noise(Wiseman & Milburn, 2010); real-world superconducting platforms often exhibit non-Markovian $1/f$ noise (Flurin et al., 2020) and calibration drifts that may require richer observation models. While our framework ensures physicality, scaling to many-qubit systems will require low-rank Kraus approximations (Ahmed et al., 2023; Ateeq et al., 2025) or geometry-aware parametrizations to manage the exponential dimensionality of the Hilbert space.

We note a distinction in inference protocols: Kraus-constrained models update the known initial state $\rho_0 = |0\rangle\langle 0|$ recursively, consistent with standard Bayesian filtering (Emzir et al., 2017; Gordon et al., 1993; Bouten et al., 2007), whereas baselines perform stateless sequence-to-state regression. Since a CPTP map defines a physical state transition, recursive propagation from ρ_0 is an architectural necessity rather than an arbitrary asymmetry. Augmenting unconstrained baselines with ρ_0 access remains a valuable direction for future work to fully isolate the fidelity contribution of the Kraus constraint. Nevertheless, the core limitation of unconstrained models remains unresolved: without strict CPTP constraints, any recursive propagation over long horizons ($T = 2000$) is susceptible to trace drift and non-physical negative eigenvalues that the Kraus layer prevents by construction.

Broader Impact. Robust state estimation is fundamental for quantum error correction and sensing(Nielsen & Chuang, 2010). Our method offers a practically deployable alternative in settings where system parameters are unknown or non-stationary, without the parameter estimation overhead of SME-based approaches or the prohibitive training cost of continuous-depth models. Models trained on synthetic data require rigorous cross-calibration before hardware deployment to prevent biased feedback control.

REFERENCES

S. Ahmed et al. Gradient-descent quantum process tomography by learning Kraus operators. *Physical Review Letters*, 130:150402, 2023. doi: 10.1103/PhysRevLett.130.150402.

Z. Ateeq et al. Geometric parameterization of Kraus operators with constraint-preserving optimization over CPTP maps. *arXiv preprint arXiv:2512.00577*, 2025. URL https://arxiv.org/abs/2512.00577.

Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018. URL https://arxiv.org/abs/1803.01271.

Luc Bouten, Ramon van Handel, and Matthew R. James. An introduction to quantum filtering. *SIAM Journal on Control and Optimization*, 46(6):2199–2241, 2007. doi: 10.1137/060651239. Published.

Heinz-Peter Breuer and Francesco Petruccione. *The Theory of Open Quantum Systems*. Oxford University Press, 2002.

Alessia Cattabriga, Elisa Ercolessi, Riccardo Gozzi, and Erika Meucci. Kraus operators and symmetric groups. *International Journal of Geometric Methods in Modern Physics*, 18(09):2150142, June 2021. ISSN 1793-6977. doi: 10.1142/s0219887821501425. URL http://dx.doi.org/10.1142/S0219887821501425.

Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. *Advances in Neural Information Processing Systems*, 2018. URL https://arxiv.org/abs/1806.07366.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014. URL https://arxiv.org/abs/1406.1078.

Muhammad F. Emzir, Matthew J. Woolley, and Ian R. Petersen. A quantum extended Kalman filter. *Journal of Physics A: Mathematical and Theoretical*, 50(22):225301, 2017. doi: 10.1088/1751-8121/aa6e5e. Published.

Emmanuel Flurin, S. Hacohen-Gourgy, L. S. Martin, and I. Siddiqi. Using a recurrent neural network to reconstruct quantum dynamics of a superconducting qubit from physical observations. *Physical Review X*, 10(1):011006, 2020. doi: 10.1103/PhysRevX.10.011006.

Neil J. Gordon, David J. Salmond, and Adrian F. M. Smith. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEE Proceedings F - Radar and Signal Processing*, 140(2): 107–113, 1993. doi: 10.1049/ip-f-2.1993.0015. Seminal particle filter paper, Published.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2024. URL https://arxiv.org/abs/2312.00752.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8): 1735–1780, 1997.

Kurt Jacobs. Quantum measurement theory and its applications. 2006.

Herbert Jaeger. The "echo state" approach to analysing and training recurrent neural networks. *GMD Report 148*, 2001.

Peter E Kloeden and Eckhard Platen. Numerical solution of stochastic differential equations. *Springer*, 1992.

Karl Kraus. *States, Effects, and Operations: Fundamental Notions of Quantum Theory*. Springer, 1983.

Neill Lambert et al. QuTiP 5: The quantum toolbox in Python. *Quantum*, 2022.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *International Conference on Learning Representations*, 2019.

H. Ma et al. Machine learning for estimation and control of quantum systems. *National Science Review*, 2025. Advance article.

K. W. Murch, S. J. Weber, K. M. Beck, E. Ginossar, and I. Siddiqi. Observing quantum state diffusion by heterodyne detection of fluorescence. *Nature*, 2013.

Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 10th anniversary edition, 2010.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. 2019.

Pierre Rouchon and Jason F. Ralph. A tutorial introduction to quantum stochastic master equations and quantum filtering. *Annual Reviews in Control*, 2022. Available via ScienceDirect.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017.

H. M. Wiseman and G. J. Milburn. *Quantum Measurement and Control*. Cambridge University Press, 2010.

A Implementation Details

To ensure an equitable and reproducible comparison, all neural architectures were allocated a parameter budget within the range of 0.3M to 1.2M trainable weights (with the exception of the reservoir-based ESN and the large-scale Transformer). Models were implemented using PyTorch 2.1 Paszke et al. (2019) and trained inside a Docker container on consumer-grade hardware to validate the accessibility of our approach for researchers without extensive computational resources.

A.1 Training Infrastructure

Hardware and Memory Constraints. All **unconstrained baseline models** were trained on an NVIDIA RTX 4050 GPU (6GB VRAM). The Kraus-constrained variants introduce additional memory overhead from the superoperator representation and QR decomposition for Stiefel manifold projection; this overhead exceeded the capacity of the RTX 4050, necessitating migration to an NVIDIA A100 (46GB VRAM) to maintain stable batch sizes and avoid gradient checkpointing artifacts. We note that the resulting differences in hardware and batch size between model classes represent limitation of the current study, and that a fully controlled comparison would require retraining all baselines on equivalent hardware; something we leave as a recommended step for any direct computational cost comparison in follow-up work.

The **Kraus-Transformer** encountered severe CUDA memory limitations even on the A100 due to the quadratic scaling of self-attention combined with the Kraus Head's operator materialization requirements. This architectural incompatibility forced a reduction in batch size to 8 and contributed to the catastrophic training instability documented in Appendix F. We suspect the memory bottleneck, combined with the fundamental mismatch between stateless global attention and the recursive Kraus update loop, was the primary cause of the Transformer's failure mode.

All models were trained with mixed-precision (FP16) using PyTorch's automatic mixed precision (`torch.cuda.amp`) to reduce memory consumption and accelerate training where applicable.

A.2 Model Architectures

Gated Recurrent Units (GRU & LSTM). The **Kraus-GRU** and **Kraus-LSTM** as well as their unconstrained **Baseline** counterparts utilize 2 hidden layers with a hidden dimension of 256. The Kraus variants map the final hidden state h_t to the Kraus Head (detailed below), while baselines use a linear projection to 8 real-valued outputs representing the flattened density matrix $\rho \in \mathbb{R}^{2\times 2}$.

Simple Recurrence (Vanilla RNN & ESN). The **Vanilla RNN** matches the GRU configuration (2 layers, hidden dimension 256) but uses standard $\tanh$ activations without gating mechanisms. The **Echo State Network (ESN)** utilizes a fixed, randomly initialized reservoir of $N = 256$ neurons with input and reservoir scaling of 0.5. Only the linear readout layer is trained using the AdamW optimizer (same as other models), not ridge regression.

Temporal Convolutional Networks (TCN). The TCN backbones consist of 4 causal residual blocks with exponentially increasing dilations $\{1, 2, 4, 8\}$ and a kernel size of 7. Both Kraus-TCN and Baseline-TCN use $d_{\text{model}} = 256$ and are evaluated in two variants: **Standard TCN** utilizes ReLU activations with dropout ($p = 0.1$), while **Gated TCN** employs a WaveNet-style $\tanh \odot \sigma$ gating mechanism to enhance temporal selectivity. The Kraus versions apply the Kraus head for CPTP updates, while the Baseline versions directly regress density matrices via a linear output layer.

Mamba (Selective State-Space). Both **Kraus-Mamba** and **Baseline-Mamba** utilize the S6 parametrization (Gu & Dao, 2024) with $d_{\text{model}} = 256$, $d_{\text{state}} = 256$, and an expansion factor of 3. We employ the optimized CUDA kernels provided in the `mamba-ssm` package, which require compute capability ≥ 8.0 (Ampere or newer).

Transformer (Global Attention). The **Kraus-Transformer** and its **baseline** utilizes 3 layers of causal masked self-attention with $d_{\text{model}} = 256$ and 4 attention heads. Positional encodings are implemented via sinusoidal embeddings. Due to memory constraints, the Kraus-Transformer was limited to a batch size of 8 and a maximum context window of $T = 2000$ timesteps.

Neural ODE (Continuous Depth). We evaluate a Neural ODE variant that utilizes a 4-layer MLP (3 hidden layers of dimension 256 with $\tanh$ activations) as the vector field function $f_\theta(\rho \oplus y)$, where $\oplus$ denotes concatenation of the flattened density matrix and measurement record. The model uses Euler integration with a normalized step size of $\Delta t = 1/T$ (where $T = 2000$ is the sequence length) over chunks of 10 measurement steps, and was trained for 100 epochs.

Output Format. All baselines output 8 real-valued numbers representing the flattened real and imaginary components of a 2×2 density matrix. No architectural constraints enforce Hermiticity, trace=1, or positive semi-definiteness. This allows us to measure physicality violations as a key evaluation metric.

A.3 THE KRAUS-STRUCTURED LAYER

The hidden representation $h_t \in \mathbb{R}^{d_h}$ is mapped to $r = 2$ Kraus operators $\{K_1, K_2\}$ per timestep; we use this compact rank in our single-qubit benchmark while enforcing trace preservation via a QR-based Stiefel constraint (up to numerical precision). The architecture proceeds as follows:

1. **Linear Projection:** A single linear layer maps h_t to 16 real-valued outputs, which are reshaped into the real and imaginary parts of a complex-valued matrix $V \in \mathbb{C}^{4\times 2}$. Specifically, the first 8 outputs form the real part and the last 8 outputs form the imaginary part.
2. **Stiefel Manifold Projection:** Apply QR decomposition to V to obtain $Q \in \mathbb{C}^{4\times 2}$ such that $Q^\dagger Q = I_2$. This enforces the completeness relation $\sum_{i=1}^{2} K_i^\dagger K_i = I$ by construction.
3. **Kraus Operator Extraction:** Reshape Q into two 2×2 matrices $\{K_1, K_2\}$.
4. **Quantum Update:** Apply the trace-renormalized Kraus map:

$$\hat{\rho}_{t+1} = \frac{\sum_{i=1}^{2} K_i \hat{\rho}_t K_i^\dagger}{\text{Tr}\left[\sum_{i=1}^{2} K_i \hat{\rho}_t K_i^\dagger\right]} \tag{7}$$

   which is implemented as standard matrix products: $K_1 \rho_t K_1^\dagger + K_2 \rho_t K_2^\dagger$. Note that the Stiefel/QR parametrization enforces $\sum_i K_i^\dagger K_i = \mathbb{I}$, so the map is theoretically trace-preserving; the explicit trace renormalization is included only to correct floating-point drift over long rollouts ($T = 2000$).

Numerical Stability. To prevent numerical instabilities in the QR decomposition, small Gaussian jitter ($\sigma = 10^{-6}$) is added to V before applying QR to avoid singularities. Trace normalization uses a stabilized denominator $\mathrm{Tr}(\rho) + 10^{-8}$ to prevent division by near-zero values during trajectory rollouts.

A.4 Hyperparameters and Training

All models were trained using the AdamW optimizer (Loshchilov & Hutter, 2019) with a weight decay of 0.01. We employed a `ReduceLROnPlateau` learning rate scheduler with a reduction factor of 0.5 and a patience of 3–5 epochs (monitoring training loss).

$$\mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} \|\rho_{\text{pred}}^{(i)} - \rho_{\text{true}}^{(i)}\|_F^2 \tag{8}$$

where N is the batch size. **Critically, no explicit physicality penalties** (trace or PSD constraints) were applied to the unconstrained baselines, allowing us to isolate the effect of the Kraus-structured output layer as a pure architectural constraint.

Table 4: Hyperparameter configurations for all architectural families.

Model	**Type**	**Hidden Dim**	**LR**	**Batch**	**Epochs**
Kraus-GRU	Kraus	256	5×10^{-4}	128	100
Kraus-LSTM	Kraus	256	10^{-3}	32	100
Kraus-RNN	Kraus	256	5×10^{-4}	32	100
Kraus-ESN	Kraus	256	5×10^{-4}	128	100
Kraus-TCN (Std)	Kraus	256	5×10^{-4}	32	100
Kraus-TCN (Gated)	Kraus	256	5×10^{-4}	32	100
Kraus-Transformer	Kraus	256	10^{-3}	8	100
Kraus-Mamba	Kraus	256	5×10^{-4}	128	100
Neural-ODE (baseline)	Baseline	256	10^{-3}	64	100
Baseline-GRU	Baseline	256	5×10^{-4}	128	100
Baseline-LSTM	Baseline	256	5×10^{-4}	128	100
Baseline-RNN	Baseline	256	5×10^{-4}	128	100
Baseline-ESN	Baseline	256	5×10^{-4}	128	100
Baseline-TCN (Std)	Baseline	256	5×10^{-4}	128	100
Baseline-TCN (Gated)	Baseline	256	5×10^{-4}	128	100
Baseline-Transformer	Baseline	256	5×10^{-4}	32	100
Baseline-Mamba	Baseline	256	5×10^{-4}	64	100

Note: Batch sizes and hidden dimensions were determined by available CUDA memory, balancing computational efficiency, hardware constraints, and training time to ensure stable convergence without out-of-memory errors.

A.5 Trainable Parameter Counts

To ensure a rigorous comparison, we report the total number of trainable parameters for each model family in Table 5. We observe that the **Kraus-GRU**, our most efficient and agile model, utilizes significantly fewer parameters (0.79M) than the **Kraus-Transformer** (3.95M) or **Kraus-TCN** (1.84M), while achieving superior adaptation performance. This suggests that the recurrent inductive bias of the GRU is better aligned with the low-rank structure of the single-qubit Hilbert space.

B Dataset Generation Details

Generative Stochastic Master Equation. The trajectories are generated using the diffusive Stochastic Master Equation (SME) under continuous homodyne monitoring of the σ_z observable. The system

Table 5: Trainable Parameter Counts for Kraus-Constrained and Baseline Architectures (All models at 256 hidden dimensions).

Kraus Models (256 Hidden Dim)	**Params**	**Baseline Models (256 Hidden Dim)**	**Params**
Kraus-Transformer (3L)	5.23M	Baseline-Transformer	2.42M
Kraus-TCN Gated	3.68M*	Baseline-TCN Gated	3.68M
Kraus-TCN Standard	1.84M	Baseline-TCN Standard	1.84M
Kraus-Mamba	1.21M	Baseline-Mamba	1.02M
Kraus-LSTM (2L)	1.06M	Baseline-LSTM	0.79M
Kraus-GRU (2L)	**0.79M**	Baseline-GRU	0.60M
Kraus-RNN (Vanilla)	0.27M	Baseline-RNN	0.20M
Neural-ODE	0.14M	—	—
Kraus-ESN	70.2k	Baseline-ESN	68.1k

evolves according to:

$$\mathrm{d}\rho_t = -\mathrm{i}[H(t), \rho_t]\mathrm{d}t + \gamma\mathcal{D}[\sigma_z](\rho_t)\mathrm{d}t + \sqrt{\gamma\eta}\mathcal{H}[\sigma_z](\rho_t)\mathrm{d}W_t \tag{9}$$

where $\mathcal{D}[L](\rho) = L\rho L^\dagger - \frac{1}{2}\{L^\dagger L, \rho\}$ is the Lindblad dissipator and $\mathcal{H}[L](\rho) = L\rho + \rho L^\dagger - \mathrm{Tr}(L\rho + \rho L^\dagger)\rho$ is the measurement back-action superoperator. We assume an ideal detector efficiency of $\eta = 1.0$.

Non-Stationary Switching Protocol. To evaluate in-context system identification, we introduce a discrete dynamical switch at a random timestep $\tau \in [400, 1600]$. During Phase 1 ($t < \tau$), the system evolves under $H_1 = \omega_1\sigma_x$. At the switch point τ, the Hamiltonian instantaneously rotates to $H_2 = \omega_2\sigma_y$ (orthogonal axis). As shown in Figure 2, this results in a sharp transition in the Bloch vector trajectories, forcing the neural filters to adapt their latent representation to the new rotation axis purely from the change in measurement statistics y_t.

Parameter Randomization. To ensure the models learn the underlying physics rather than memorizing a fixed parameter set, the Rabi frequencies and measurement strengths are uniformly sampled per trajectory:

- Phase 1 Rabi frequency: $\omega_1 \sim \mathcal{U}[0.5, 4.0]$
- Phase 2 Rabi frequency: $\omega_2 \sim \mathcal{U}[0.5, 4.0]$
- Measurement strength: $\gamma \sim \mathcal{U}[0.3, 0.8]$ (held constant across both phases)

Numerical Integration. Trajectories are simulated using QuTiP 5.x with the Euler-Maruyama method for stochastic differential equations Kloeden & Platen (1992). The integration time step is $\Delta t = 0.005$, and each trajectory consists of $T = 2000$ timesteps (total physical time 10 seconds). The initial state is set to $\rho_0 = |0\rangle\langle 0|$ (ground state) for all trajectories.

Standardization. The raw homodyne current dy_t is highly stochastic and exhibits high variance. To stabilize training, we perform global standardization using the training set statistics:

$$y'_t = \frac{y_t - \mu}{\sigma + 10^{-8}} \tag{10}$$

where μ and σ are the mean and standard deviation computed over all training trajectories.

Train/Test Splits. To ensure rigorous generalization assessment, the dataset is partitioned into non-overlapping trajectory sets using distinct random seeds for simulation. The training set (2000 trajectories) is generated with `numpy.random.seed(45)`, while the test set (300 trajectories) uses `seed(9999)`. This guarantees that test trajectories exhibit statistically independent realizations of Wiener noise, parameter combinations (ω, γ), and switching times τ, preventing any form of temporal or statistical leakage between splits. All reported test metrics reflect true out-of-sample performance.

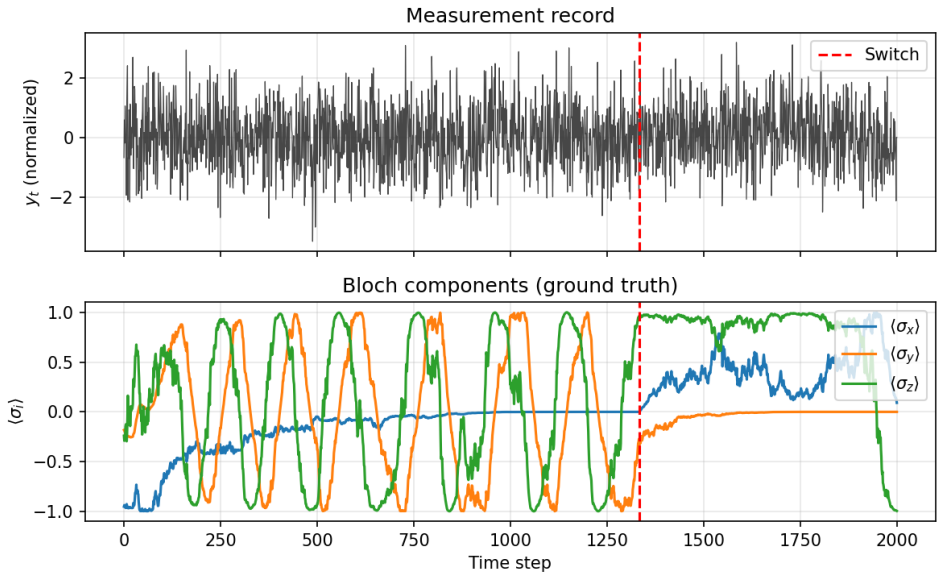

Figure 2: **Visualization of the Switching Dataset.** (Top) The normalized homodyne measurement record y_t provided as input to the models. (Bottom) Ground-truth Bloch vector components showing the Phase 1 (σ_x-rotation) to Phase 2 (σ_y-rotation) transition at the vertical dashed line. The stochastic "jitter" in the Bloch components illustrates the measurement back-action (kicks) modeled by the SME.

C Evaluation Metrics

We utilize a set of physics-motivated metrics to evaluate both reconstruction accuracy and physical consistency.

Fidelity and Bures Distance. The primary accuracy metric is the state fidelity:

$$F(\rho, \sigma) = \left(\mathrm{Tr}\sqrt{\sqrt{\rho}\sigma\sqrt{\rho}}\right)^2 \tag{11}$$

For our conditioned trajectories (which remain nearly pure), we utilize the product trace $\mathrm{Tr}(\rho\sigma)$ as a computationally efficient proxy. To characterize the distance on the quantum state manifold, we report the Bures distance:

$$d_B(\rho, \sigma) = \sqrt{2(1 - \sqrt{F(\rho, \sigma)})} \tag{12}$$

Physicality Violation Metrics. We quantify the violation of physical constraints using absolute error measures, consistent with our implementation:

- **Trace Error (V_{tr}):** Calculated as the mean absolute deviation from unit trace:

$$V_{\mathrm{tr}} = \langle|\mathrm{Tr}(\hat{\rho}_t) - 1|\rangle \tag{13}$$

- **Positivity Violation (V_{PSD}):** Defined as the magnitude of the most negative eigenvalue, ensuring the state remains within the positive semi-definite cone:

$$V_{\mathrm{PSD}} = \langle\max(0, -\lambda_{\min}(\hat{\rho}_t))\rangle \tag{14}$$

- **Hermiticity Error (V_{herm}):** Measures the deviation from Hermitian symmetry:

$$V_{\mathrm{herm}} = \langle\|\hat{\rho}_t - \hat{\rho}_t^\dagger\|_F\rangle \tag{15}$$

In our benchmark, a model is classified as Physical only if the maximum trace deviation satisfies $V_{\mathrm{tr}} = \max_t |\mathrm{Tr}(\rho_t) - 1| < 10^{-4}$ and the minimum eigenvalue satisfies $\lambda_{\min}(\rho_t) \geq -10^{-6}$ across the entire test ensemble. These tolerances are chosen to be tight relative to single-precision numerical error, ensuring strict practical physicality.

Additional Metrics. For completeness, we also report:

- **Frobenius Norm:** $\|\hat{\rho}_t - \rho_t\|_F$ for direct state reconstruction error.
- **Bloch Vector Error:** For single-qubit systems, we compute the Euclidean distance between predicted and true Bloch vectors: $\|\vec{r}_{\text{pred}} - \vec{r}_{\text{true}}\|_2$, where $\vec{r} = \text{Tr}(\vec{\sigma}\rho)$.

D PHYSICS-BASED SME BASELINE

The Adaptive SME serves as a physics-based reference that directly integrates the Stochastic Master Equation using QuTiP 5.2 (Lambert et al., 2022). Unlike the learned neural filters, this approach applies the theoretical SME dynamics with online parameter estimation, requiring no training.

SME Integration. The filter implements the standard diffusive SME for homodyne measurement:

$$d\rho_t = -i[H(t), \rho_t]dt + \gamma\mathcal{D}[\sigma_z](\rho_t)dt + \sqrt{\gamma}\mathcal{H}[\sigma_z](\rho_t)dW_t \tag{16}$$

where $\mathcal{D}[L](\rho) = L\rho L^\dagger - \frac{1}{2}\{L^\dagger L, \rho\}$ is the Lindblad dissipator and $\mathcal{H}[L](\rho) = L\rho + \rho L^\dagger - \text{Tr}(L\rho + \rho L^\dagger)\rho$ is the measurement back-action superoperator. Integration uses Euler-Maruyama with $\Delta t = 0.005$.

Adaptive Parameter Estimation. Since the true system parameters (ω, γ) are unknown and change at the switching point, the Adaptive SME performs online estimation: (i) **Rabi frequency** ω is estimated via sliding-window FFT (window size 100 steps) on the measurement record, (ii) **Measurement strength** γ is estimated from measurement variance and clipped to $[0.1, 2.0]$, and (iii) **Switch detection** monitors the innovation $(y_t - \sqrt{\gamma}\langle L + L^\dagger\rangle)^2$ for abrupt increases and resets the estimation window when a regime change is detected.

Performance and Analysis. The Adaptive SME achieves a mean fidelity of 0.6780 on the switching test set (Bures distance: 0.4318, Phase 1: 0.6825, Phase 2: 0.6705). This is competitive with several neural baselines but falls short of the best Kraus-constrained models (Kraus-LSTM: 0.7683, Kraus-GRU: 0.7655).

This performance gap is *not* a fundamental limitation of the SME formalism itself—in ideal conditions with raw homodyne currents and known system parameters, the SME can achieve $\sim$95% fidelity agreement with quantum state tomography (Murch et al., 2013). Rather, the degradation arises from two compounding factors that are *intentional design choices* in our benchmark to ensure the dataset poses a realistic challenge for adaptive filtering:

1. **Parameter estimation error**: The Adaptive SME must estimate both ω (via sliding-window FFT) and γ (via variance) from noisy measurements. These estimates are inherently imperfect, particularly during transient dynamics immediately after the regime switch when the estimation window contains mixed statistics from both phases. The fact that even a well-tuned physics-based filter with explicit parameter estimation struggles on this dataset validates that the switching dynamics present a non-trivial system identification challenge, rather than a task solvable by simple pattern memorization.
2. **Switch detection latency**: The innovation-based switch detector requires $\sim$50 timesteps to accumulate sufficient evidence of a regime change before resetting the parameter estimation window. During this latency period, the filter applies Phase 1 parameters (ω_1, $H \propto \sigma_x$) to Phase 2 dynamics (ω_2, $H \propto \sigma_y$), leading to transient tracking errors that degrade the overall fidelity.

In contrast, the learned Kraus-constrained models implicitly encode the filtering dynamics in their weights and can adapt to regime changes within a few timesteps via their recurrent hidden states, without requiring explicit parameter re-estimation. This demonstrates that end-to-end learning can outperform physics-based approaches when system parameters are unknown and non-stationary, even when the learned models enforce the same CPTP structure as the theoretical SME.

E VANILLA RNN GRADIENT DYNAMICS UNDER KRAUS CONSTRAINTS

We investigate the marginal fidelity degradation observed in the Vanilla RNN under Kraus constraints (Table 2, $\Delta = -0.0036$) by examining layer-wise gradient norms computed from a forward-backward pass on the test set using the final trained checkpoint.

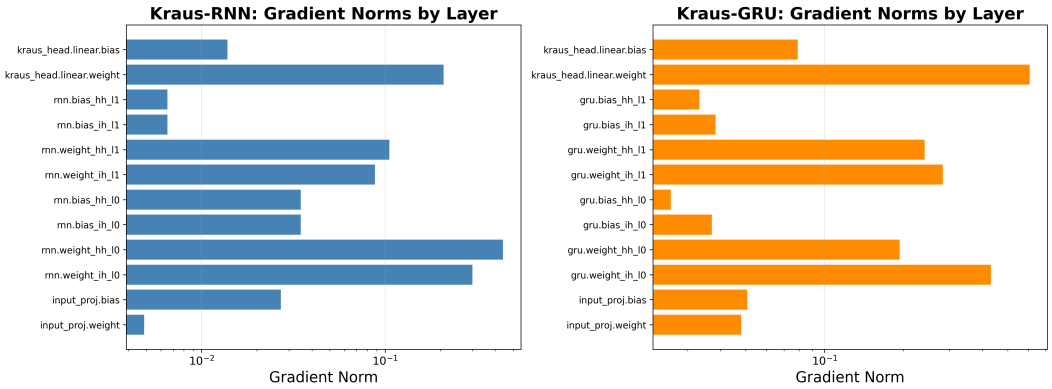

Figure 3: Layer-wise gradient norms for Kraus-RNN (left) and Kraus-GRU (right) computed from trained checkpoints.

As shown in Figure 3,the Kraus-RNN exhibits severe gradient imbalance across layers: the input projection weights receive nearly vanishing gradients ($\sim 10^{-2.5}$), while recurrent weights at later layers (`rnn.weight_hh_l0`) receive gradients an order of magnitude larger. This imbalance suggests that early-layer representations are effectively frozen during training, preventing the model from learning input-responsive dynamics. In contrast, Kraus-GRU maintains comparatively uniform gradient flow, with all layers receiving gradients of similar magnitude ($\sim 10^{-1}$).

We hypothesize that this gradient imbalance arises from the interaction between two compounding factors: (i) the well-known vanishing gradient problem in ungated recurrence over long sequences ($T = 2000$), and (ii) the QR-based Stiefel projection in the Kraus head, which constrains the effective update directions available to the optimizer. Together, these factors render the Kraus-RNN overly inertial after abrupt regime changes, as the frozen input projection cannot rapidly re-encode post-switch measurement statistics into the recurrent hidden state. Gated architectures avoid this pathology by decoupling input sensitivity from hidden state retention via reset and forget gates.

We note that this remains a hypothesis consistent with the observed gradient structure, rather than a definitive causal explanation. A more controlled ablation, such as replacing the QR parametrization with an unconstrained Kraus head in the RNN, or applying gradient clipping, is left to future work.

F FAILURE ANALYSIS

A significant outcome of our benchmark is the performance collapse of the Kraus-Transformer, which achieved a mean fidelity of 0.5416 ,representing a catastrophic -0.2030 **degradation** relative to its unconstrained baseline (Baseline-Transformer: 0.7446). This is the only architecture in our study where Kraus constraints severely harm performance, revealing a fundamental incompatibility between stateless attention mechanisms and recursive physics-informed layers.

Manifold Collapse and Recursive Drift. We observe that the Kraus-Transformer suffers from manifold collapse toward the center of the Bloch sphere. Because the Kraus update ($\rho_{t+1} = \mathcal{K}_t[\rho_t]$) is an iterative process, each operator K_t must be perfectly aligned with the current phase of the state. Since the Transformer is stateless and lacks an explicit hidden memory to track the density matrix's evolution, it fails to maintain phase coherence. Consequently, the predicted Kraus operators act as a stochastic drain, causing the state to spiral inward toward the maximally mixed state ($\rho = I/2$). This results in a trajectory that resembles a random walk confined to the center of the Hilbert space, explaining the 0.54 fidelity, which represents the baseline of an uninformative prediction.

Incompatibility with Markovian SDEs. These results confirm that global self-attention is an improper inductive bias for Markovian Stochastic Differential Equations. By attending to a broad window of past noise increments without a persistent latent anchor, the Transformer introduces look-back interference that disrupts the sequential integration required by the Kraus Representation Theorem. Unlike the GRU and Mamba architectures, which maintain an evolving hidden state h_t that successfully emulates the conditional density matrix, the Transformer is unable to provide the local context necessary to drive a stable recursive physical map.

Memory Constraints and Training Instability. As documented in Appendix A, the Kraus-Transformer encountered severe CUDA memory limitations even on the A100 GPU, forcing a reduction in batch size to 8. This small batch size contributed to noisy gradient estimates during training, exacerbating the architectural incompatibilities described above. The combination of stateless attention, recursive error accumulation, and memory-constrained training created a catastrophic failure mode unique to this architecture.

Final note The Kraus layer acts as a rigorous stress test for architectural inductive biases. Architectures that excel at direct state regression (curve-fitting) do not necessarily succeed when constrained to operate within the CPTP manifold through recursive updates. Our results demonstrate that successful quantum filtering requires explicit recurrent statefulness to maintain temporal coherence across the iterative Kraus map.