A Time-Regulated Neural Operator Framework for Chaotic Biomedical System Forecasting

Aarjav Jain[1], Sai Hruday Reddy Nara[1], Alden Raymond[1], Karthik Natraj[1], Cyrus Xue[1], Marx Akl**

[1]*Department of Computer Science and Engineering, Aspiring Scholars Directed Research Program, Fremont, California 94539 USA*

***Corresponding author*

ARTICLE INFO

Keywords:

CPU COMPUTING POWER

TIME TRAINED

ABSTRACT

Chaotic dynamical systems pose a fundamental challenge for data-driven prediction because small deviations in initial conditions lead to exponentially diverging trajectories. Existing neural operator approaches, such as the Fourier Neural Operator and DeepONet, learn function-to-function mappings and are particularly useful for higher-order differential equations; however, they tend to be computationally expensive for lower-dimensional chaotic DEs. In this study, we propose an alternative method with a time-step–indexed operator learning framework that restructures the prediction problem to learn a sequence of localized transition operators, each modeling short-horizon flow maps conditioned on the system's evolving latent state rather than learning a single global operator. This formulation blends ideas from neural operators, recurrent

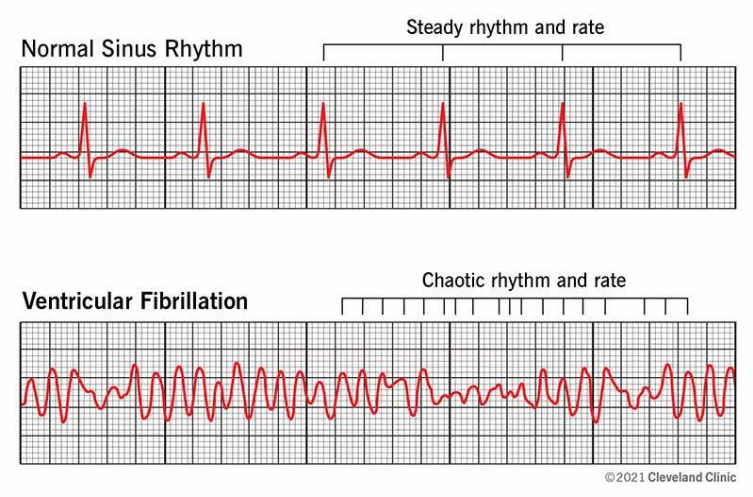
state-space modeling, and reservoir-based stability mechanisms, enabling long-window forecasting without conventional autoregressive drift and is efficient for lower dimensional ODEs.

To evaluate our approach, we perform extensive experiments on the Mackey–Glass and double pendulum systems. Quantitatively, our method reduces long-horizon prediction error relative to DeepONet and FNO by ____% and ____%, respectively, on trajectories of length 200. The largest Lyapunov exponent estimated from predicted trajectories differs from the ground truth by only ____, compared to ____ for baseline operator models, demonstrating significantly improved reproduction of chaotic divergence rates. Additionally, our model maintains stable rollouts for 104 time steps before deviation, outperforming autoregressive LSTM baselines by a factor of 100.

These results provide the first evidence that a time-indexed operator decomposition can overcome core limitations of global neural operators in chaotic regimes. Beyond improving accuracy, our method offers a principled pathway toward operator learning architectures that explicitly respect local flow structure, making them suitable for scientific forecasting, control, and system identification in highly sensitive dynamical systems.

I N T R O D U C T I O N

Chaotic systems are inherently sensitive to initial conditions, exhibiting strong instabilities where small perturbations amplify exponentially over time due to positive Lyapunov exponents. This intrinsic sensitivity makes both analytical and numerical prediction of long-term trajectories extremely challenging, as even minimal errors accumulate rapidly, rendering precise forecasting infeasible beyond short time horizons. [] Specifically, accurately modeling and predicting biological chaotic systems is one of the foremost challenges in contemporary computational biology. Many biological processes—ranging from hematopoietic dynamics and hormone regulation to neural oscillations and cardiac rhythms—exhibit intrinsic nonlinear and delayed feedback mechanisms. These mechanisms often induce chaotic behavior, characterized by sensitive dependence on initial conditions, high-dimensional nonlinear interactions, and complex temporal patterns. Such properties make long-term prediction fundamentally difficult: even minuscule perturbations in a system’s initial state can propagate exponentially, leading to dramatically divergent outcomes over time. []



https://my.clevelandclinic.org/health/diseases/21878-ventricular-fibrillation

Traditional biological modeling relies on mechanistic or rule-based simulations, often encoded as ordinary differential equations (ODEs) or delay differential equations (DDEs), such as the classical Mackey–Glass system. While these models capture essential system dynamics and can reproduce qualitative behavior, they face significant limitations. Analytical solutions are generally infeasible, numerical solutions are computationally expensive and error-prone for long-term simulations, and parameter tuning requires extensive domain expertise. Consequently, the practical application of these models for predictive, patient-specific, or real-time interventions is severely constrained.

In recent years, machine learning (ML) approaches have emerged as powerful tools for modeling chaotic systems. Recurrent neural networks (RNNs), Long Short-Term Memory networks (LSTMs), and Gated Recurrent Units (GRUs) have been applied successfully to low-dimensional chaotic systems, such as the Lorenz-63 attractor and Mackey–Glass time series. Traditional approaches for modeling chaotic dynamics have therefore focused on short-term predictions, often training on brief trajectory segments or step-wise projections. These methods typically use sliding-window or stepwise prediction strategies, training the model on short trajectory segments and predicting the immediate next state. While effective over short horizons, these approaches suffer from two fundamental limitations:

1. Trajectory Dependence and Overfitting: Sliding-window models are often tightly coupled to specific initial conditions, failing to generalize to new trajectories or different parameter regimes.

2. Error Accumulation and Predictability Horizon: Due to the exponential divergence inherent in chaotic systems (quantified by positive Lyapunov exponents), sequential predictions accumulate error rapidly, severely limiting long-term forecasting.

Alternative methods, such as the Frobenius-Perron operator and the Koopman operator frameworks, attempt to capture the evolution of probability densities or observables in the system's state space. While theoretically appealing, these operator-based methods require linear approximation in function spaces, which becomes particularly challenging in high-dimensional or infinite-dimensional systems. Consequently, prior approaches succeed primarily for low-dimensional ordinary differential equations (ODEs), such as the Lorenz-63 system, or one-dimensional partial differential equations (PDEs) like the Kuramoto-Sivashinsky equation, but fail to capture long-term behavior in more complex systems, including turbulent flows governed by Navier-Stokes dynamics.

Given these limitations, a promising alternative paradigm — neural operators — has gained traction, especially in physics and engineering domains. In neural operator learning, one seeks to approximate the solution operator of a dynamical system: a functional mapping that takes an initial condition (or input function) and returns the full future evolution (or solution at a target time), rather than step-by-step predictions. Pioneering work on Neural Operator (NO) demonstrated that neural networks can be extended from finite-dimensional mappings to function-to-function maps in infinite-dimensional spaces, with universal approximation guarantees and discretization invariance. Subsequent architectures such as U-NO and other variants (e.g. graph, low-rank, Fourier) improved memory efficiency, stability, and handling of complex geometries.

Neural operators have achieved impressive performance on partial differential equations (PDEs), including the Navier–Stokes equations, Darcy flow, and other physical systems, delivering orders-of-magnitude speedups over classical solvers while maintaining accuracy. However, the literature remains heavily focused on spatial PDEs, or time-dependent PDEs with smooth, well-behaved dynamics; there is little to no prior work applying neural-operator methods to chaotic ODEs or delay differential equations with biological relevance. In particular, operator-learning frameworks have seldom addressed delay-induced chaos or time-delay feedback systems, which pose unique challenges: non-smooth dependence on history, infinite-dimensional state due to delays, and extreme sensitivity to initial conditions.

This gap motivates our present study. In this study, we introduce a time-step-based operator learning framework that addresses these limitations. Our approach diverges fundamentally from classical sliding-window methods: rather than predicting the next immediate state based on historical trajectory segments, we train a neural operator to map any initial condition directly to a future state at an arbitrary time horizon.

Our method diverges from classical sliding-window ML and from standard operator-learning PDE applications in several critical ways:

- We learn a direct mapping from initial condition to future state at arbitrary target time (t0→t0+Δtt_0 \rightarrow t_0 + \Delta tt0→t0+Δt), rather than iterating small time-step predictions or relying on trajectory history.

- We train over a diverse distribution of initial conditions, including out-of-distribution and perturbed states, to encourage true generalization rather than trajectory memorization.

- We incorporate stability-aware loss design and architectural choices to respect dissipativity and mitigate divergence under chaotic dynamics, which are not addressed in typical neural-operator PDE literature.

- We demonstrate robustness and generalization across parameter regimes and initial states, indicating potential applicability to real biological delayed-feedback systems beyond synthetic benchmarks.

By explicitly modeling the functional dependence between initial states and future outcomes, our method achieves generalization across initial conditions and allows compositional long-term forecasting through repeated operator application. To further ensure biological realism, we incorporate dissipativity constraints, guaranteeing that predicted trajectories remain confined to the system's global attractor, preserving the statistical and dynamical properties of the underlying chaotic system.

Using the Mackey–Glass delay differential equation as a canonical testbed, our experiments show that the learned operator achieves high accuracy for long-term forecasting, outperforms recurrent/sliding-window baselines, and generalizes to previously unseen initial conditions and regimes — a combination rarely demonstrated in prior literature.

Beyond its methodological contributions, our work provides a conceptual framework for modeling chaotic biological systems using operator learning. By bridging machine learning, chaos theory, and computational biology, this approach opens the possibility of patient-specific forecasting, disease progression modeling, and intervention planning, potentially impacting hematology, cardiology, and neuroscience. In summary, this study presents a first time-step-based operator learning framework tailored for delayed biological chaos modeled by ODEs, providing a principled and scalable solution for predictive modeling in complex biological systems.

2. Methodology

2.0 Mackey-Glass Dynamics

The Mackey-Glass (MG) system is a canonical delay-differential equation (DDE) widely employed in the study of nonlinear dynamics and chaos, particularly in biological contexts [cite papers in comment]. Originally proposed to model the feedback regulation of blood cell production, the MG system has become a benchmark example in chaos theory due to its capacity to exhibit a wide spectrum of dynamical behaviors—from fixed points to periodic oscillations and fully developed chaos. DDEs, in contrast to ordinary differential equations, explicitly account for time delays inherent in many natural and biological systems, such as maturation periods, transport lags, and delayed feedback in regulatory pathways.

The general form of the Mackey-Glass equation is:

$$\frac{dx(t)}{dt} = \beta \cdot \frac{x(t-\tau)}{1 + x(t-\tau)^n} - \gamma \cdot x(t)$$

Where:

- $x(t) \in$ (all real) represents the system state at time t, such as a normalized cell concentration.
- $\beta > 0$ is the production rate coefficient.
- $\gamma > 0$is the decay rate coefficient.
- $n > 0$is a nonlinearity parameter controlling the saturation of feedback.
- $\tau > 0$ is the critical delay term, representing the time lag between the current state and its influence on the system derivative. Biologically, τ models delays such as the time for precursor blood cells to mature before entering circulation.

On the other hand, the multi-pendulum (MP) system is an example of a coupled nonlinear dynamical system widely used in the study of classical mechanics, nonlinear mechanics, nonlinear dynamics, and chaos theory. Consisting of two or more pendulums connected through joints, the system is commonly investigated to understand the dynamics of energy transfer and stability in mechanical systems. Nevertheless, it is now a benchmark model for exploring and predicting complex behavior resulting from simple deterministic laws. The combination between pendulums introduces nonlinear interactions, leading to phenomena such as energy localization, resonance, and chaotic phase-space trajectories. These properties make the system a valuable testbed for investigating chaos and the limits of predictability in deterministic systems.

The system's qualitative behavior depends sensitively on τ. For small τ, the system converges to stable fixed points. Intermediate values of τ produce periodic oscillations, while sufficiently large τ values lead to chaotic dynamics, characterized by sensitive dependence on initial conditions []. In this study, we selected τ=17, which is well-known to produce chaotic trajectories under standard parameter settings (β=0.2, γ=0.1, n=10), and an array of discrete initial conditions separated by broken intervals of 0-1.17 on the left and 1.23-3 on the right (elaboration for this choice in 2.4). All trajectories were generated for $t \geq 0$, and points were taken at intervals of Δt=1. While smaller timesteps are often used for higher numerical resolution, reducing Δt below 1 in our study produces a vastly larger number of points without adding new information about the system dynamics. The underlying system is continuous, so trajectories generated with smaller timesteps are effectively the same as those with Δt = 1; they simply contain more densely sampled points. Conceptually, this choice does not reduce the information content — it merely reduces redundant temporal samples, preserving all relevant dynamic features of the system.

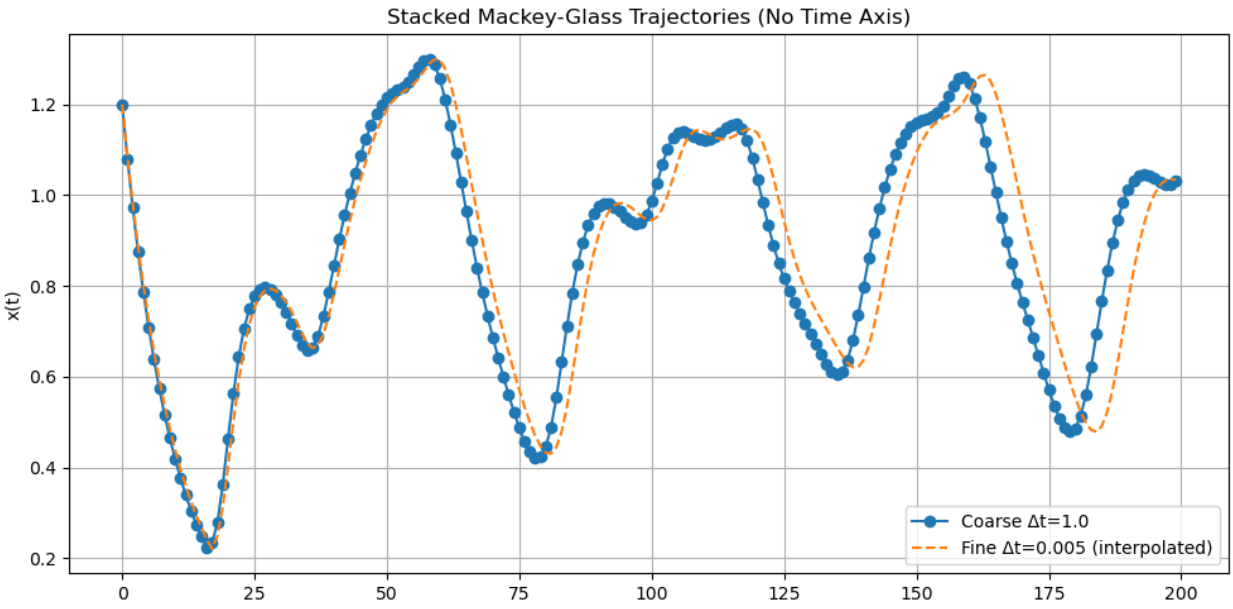


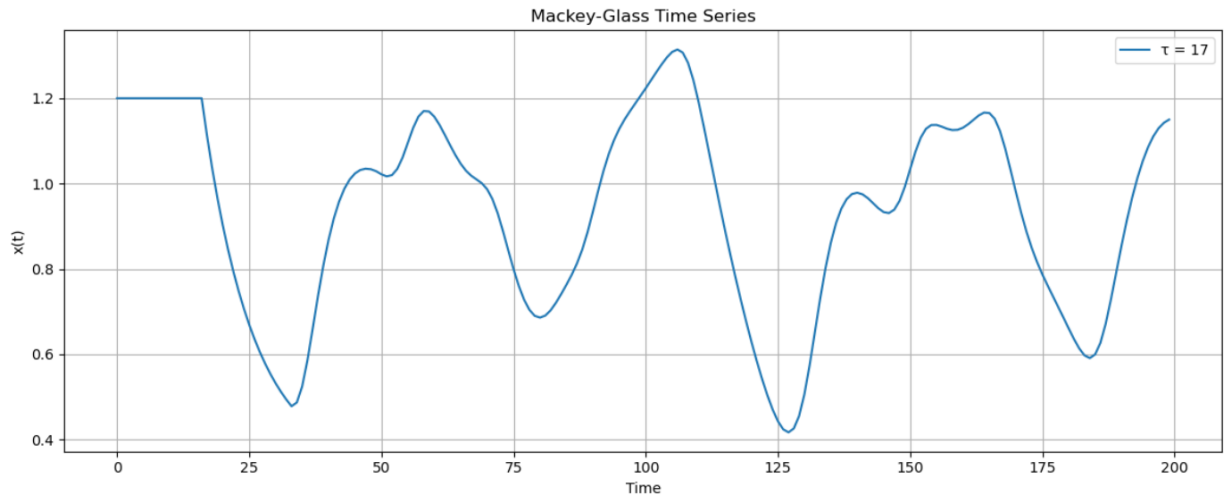
Figure X: *Comparison of Mackey-Glass trajectories generated with different timesteps. The coarse trajectory (Δt = 1) is plotted in same graph as the fine trajectory (Δt = 0.005) after interpolation to match the coarse points. Despite the difference in sampling resolution, the interpolated fine trajectory closely aligns with the coarse trajectory, demonstrating that reducing Δt increases temporal resolution but does not change the underlying dynamics.*

Step	Coarse ($\Delta t = 1$)	Fine ($\Delta t = 0.005$)	Interpolated Fine
1	1.2000	1.2000	1.2000
2	1.0800	1.1994	1.0852
3	0.9720	1.1988	0.9814
4	0.8748	1.1982	0.8876
5	0.7873	1.1976	0.8027

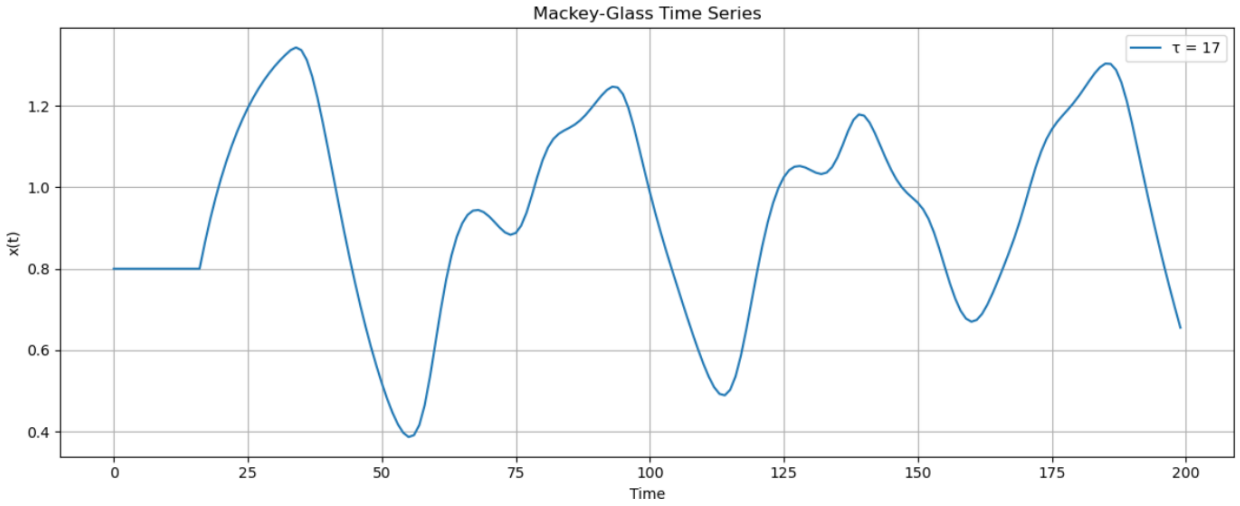
Figure X: Comparison of Mackey-Glass trajectories generated with different timesteps (table form with first 5 concentration values). The coarse trajectory (Δt = 1) has discrete differences between steps, is sufficient for the models to learn the underlying dynamics of the system. With the fine trajectory (Δt = 0.005), the first 5 points are merely minor perturbations that creates redundant temporal samples. After interpolation, the pattern and values for the fine points with match the coarse points given by the table.

To generate MG trajectories, we implemented a custom Python Euler integration scheme, allowing precise control over system parameters, initial conditions, and anomaly insertion (2.0.2). This approach ensured reproducibility and minimized dependencies on external packages. For each trajectory, the system was integrated over t=200 discrete time steps, producing a high-resolution representation of its chaotic evolution.

A)



B)



C)

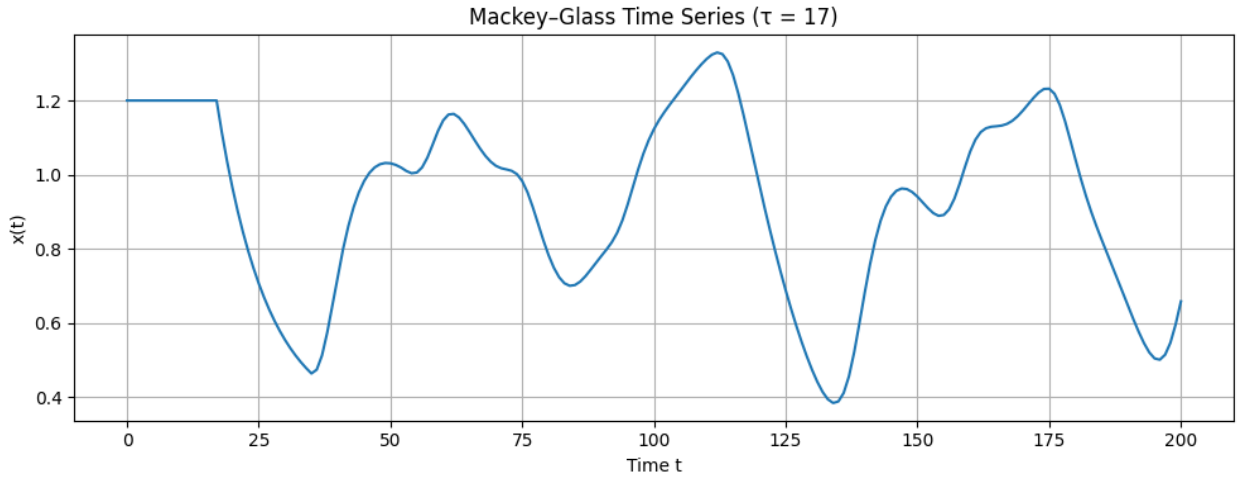


Figure X. *Graphs of the Mackey-Glass system for 200 timesteps starting at varying initial conditions*
(A) *The Mackey-Glass system starting at the initial condition 1.2.*
(B) *The Mackey-Glass system starting at the initial condition 0.8.*
(C) *The Mackey-Glass system starting at the initial condition 1.6.*

2.0.1 Lyapunov Exponents

A fundamental metric for characterizing chaos is the Lyapunov exponent (λ), which quantifies the system's sensitivity to infinitesimal perturbations in the initial condition. Formally, for a small perturbation δx(0) in the initial state, the divergence over time is measured as:

$$\lambda = \lim_{n\to\infty} \left\{ \frac{1}{n} \sum_{i=0}^{n-1} \ln \left| f'(x_i) \right| \right\}$$

Figure X: *Lyapunov Exponent Calculation*

A positive λ indicates exponential divergence of initially close states and confirms chaotic behavior, whereas zero or negative values indicate periodic or stable dynamics. In our simulations with τ=17, the MG system produced a positive Lyapunov exponent, quantitatively confirming its chaotic regime. This metric provides a foundation for evaluating model performance and understanding predictability limits, particularly relevant for models trained under the proposed time-step generalization framework.

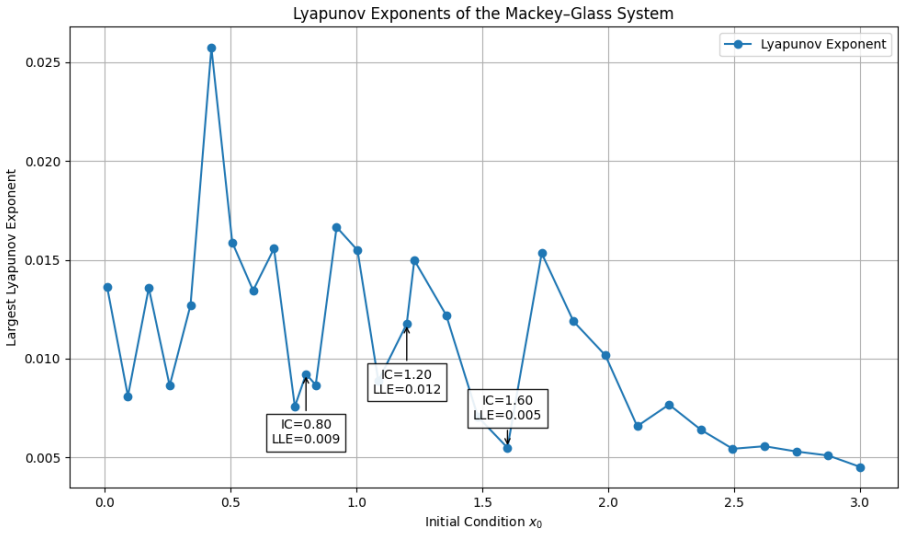


Figure X: *Lyapunov Exponent for Mackey-Glass system at tau = 17 for various ICs. Since our Lyapunov exponent is always greater than 0, the Mackey-Glass system qualifies as chaotic.*

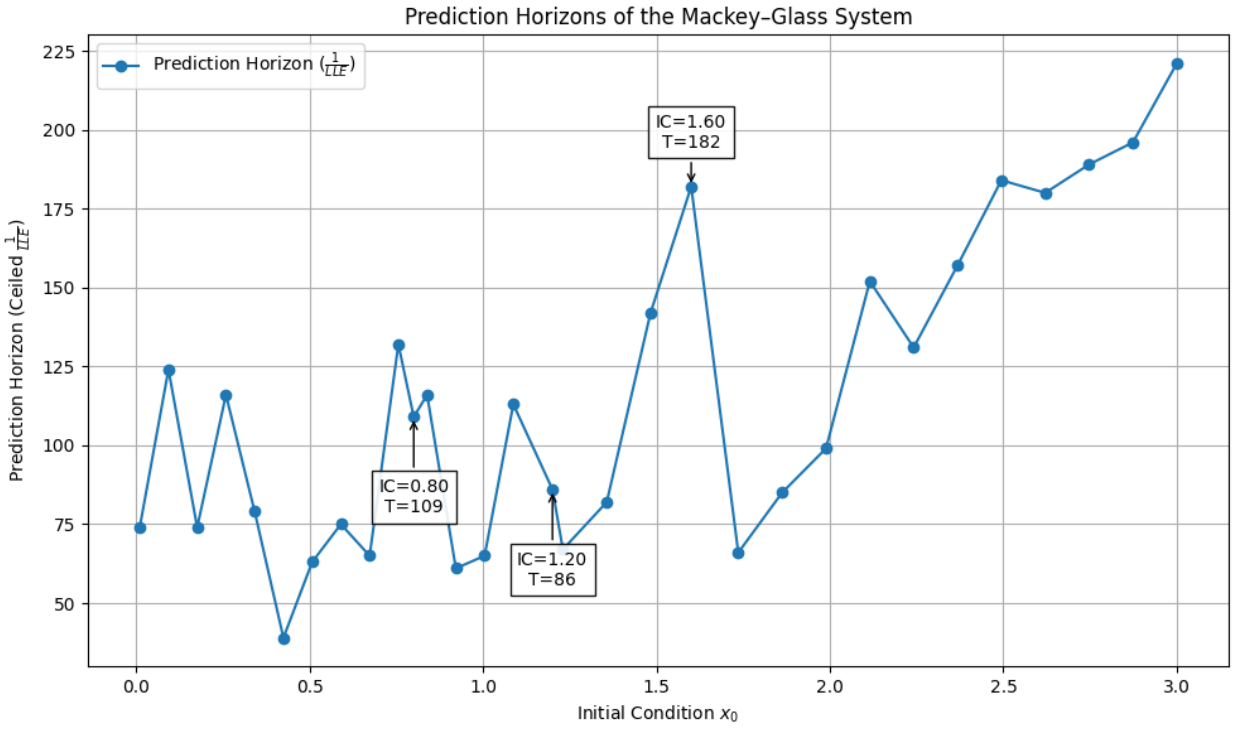


Figure X: *Lyapunov Times for Mackey-Glass system at tau = 17 for various ICs. We attempt to predict at upwards of 2 times the Lyapunov times.*

We calculate the Lyapunov Exponent with the Rosenstein method. Using the hyperparameters tau=17, delta_t=1, beta=0.2, gamma=0.1, n=10, x0=1.2, we reconstructed the system's phase space using a delay embedding approach, allowing the trajectory to be represented in a higher-dimensional space. The Largest Lyapunov Exponent (LLE) was estimated using the Rosenstein method, which measures the exponential divergence of nearby trajectories in this reconstructed space. Higher LLE values correspond to regions where small differences in ICs lead to rapidly diverging trajectories to limit the time horizon over which accurate predictions can be made. The time horizon (shown in Figure __) is quantified as approximately 1/LLE, where it is a measure of how long the system can be accurately forecasted from a given initial state.

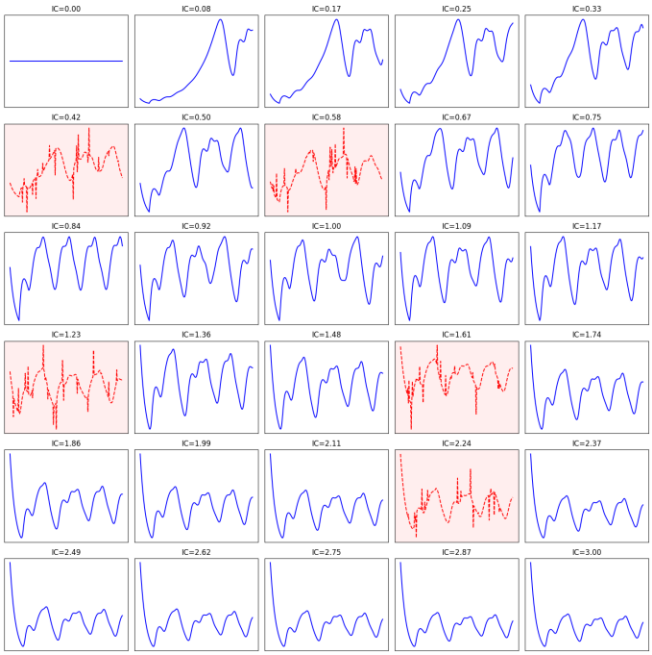
2.0.2 Anomaly Injection in Mackey-Glass Trajectories

To test model robustness under realistic perturbations, we introduced synthetic anomalies into a subset of training trajectories. In biological systems, sudden perturbations can occur due to environmental stimuli, measurement noise, or rare events. To simulate these conditions, anomalies were injected randomly at 10% of time steps within 5 of the 40 training trajectories.

Each anomaly was defined as a uniform random deviation:

$x(t)' = x(t) + \delta_t$,

where $\delta_t = 0.7$ represents the anomaly scale. This sparse and isolated anomaly design preserved the underlying chaotic dynamics while providing a controlled test of model robustness. The held-out trajectory (x0=1.2, not displayed on graph) remained anomaly-free, allowing evaluation of cross-trajectory generalization in unseen initial conditions.



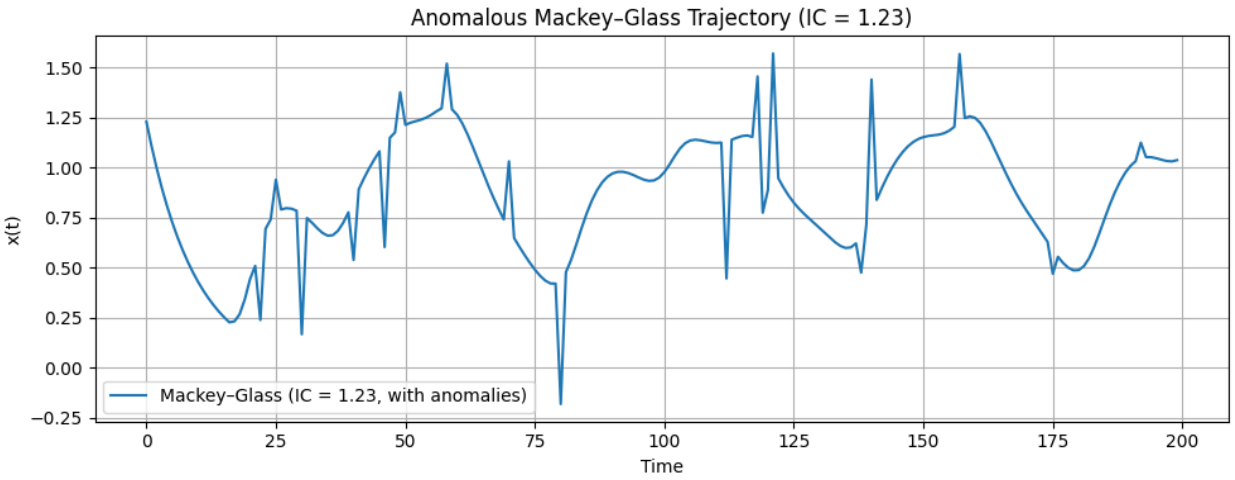
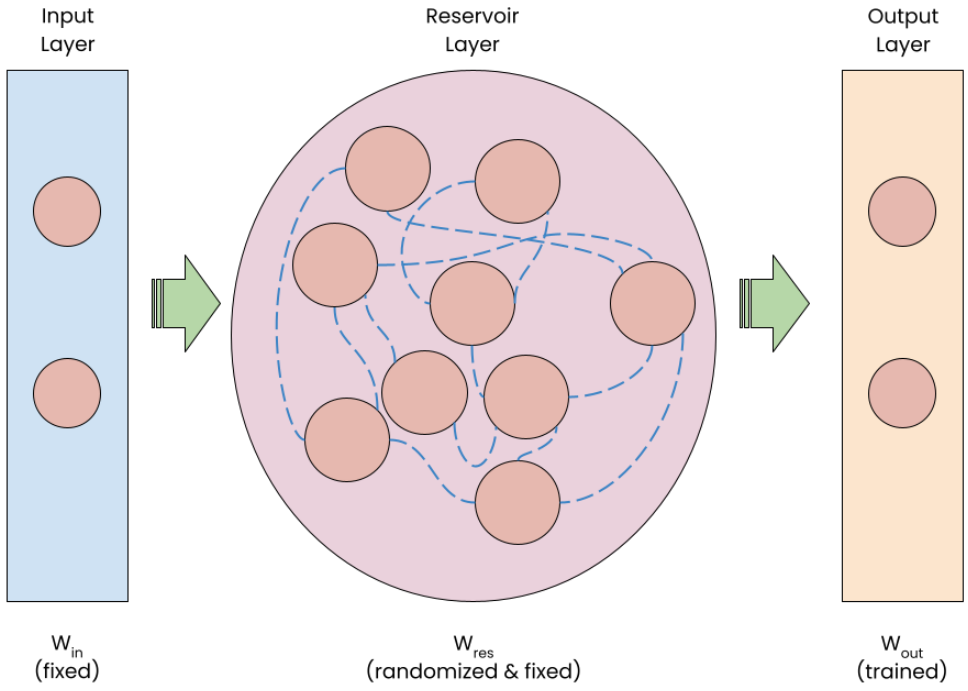


Figure X: *Sample training dataset with synthetic anomaly insertion. In our dataset, to simulate significant perturbations in trajectories with a high degree of correlation with our test trajectory (IC= 1.2), we intentionally randomized until synthetic anomalies occurred in trajectories near our held-out IC. (e.g. IC=1.23).*

2.1 Reservoir Computing

Reservoir Computing (RC) is a computational framework well-suited for modeling complex, temporal dynamics such as the Mackey-Glass system. Echo State Networks (ESNs) are a widely adopted RC implementation, consisting of a fixed, randomly connected “reservoir” of neurons. The reservoir transforms the input sequence into a high-dimensional dynamic representation, which is then linearly combined at the output layer for prediction.

The reservoir’s nonlinear dynamics allow it to capture rich temporal patterns, while the lightweight training of the output layer ensures computational efficiency. The reservoir responds dynamically to changing inputs without altering the fixed internal weights, allowing it to model complex, time-dependent chaotic systems both efficiently and effectively. ESNs therefore provide a strong baseline for predicting the MG system under our time-step-based input framework.



2.2 Nonlinear State-Space Models

Nonlinear state-space models (SSMs) provide a principled framework for modeling dynamical systems whose evolution is governed by latent (unobserved) states and nonlinear interactions. Unlike traditional sequence models that predict observations directly, SSMs assume the existence of underlying hidden states z_t that evolve according to a nonlinear dynamical system and give rise to observed measurements y_t. This formulation is particularly advantageous for

chaotic biological systems, such as Mackey-Glass, where the measured signal reflects only partial information of the system's internal state.

This framework decouples the latent dynamics from the stochasticity or partial observability of measurements, allowing the model to capture the intrinsic structure of chaotic systems while accounting for measurement uncertainty.

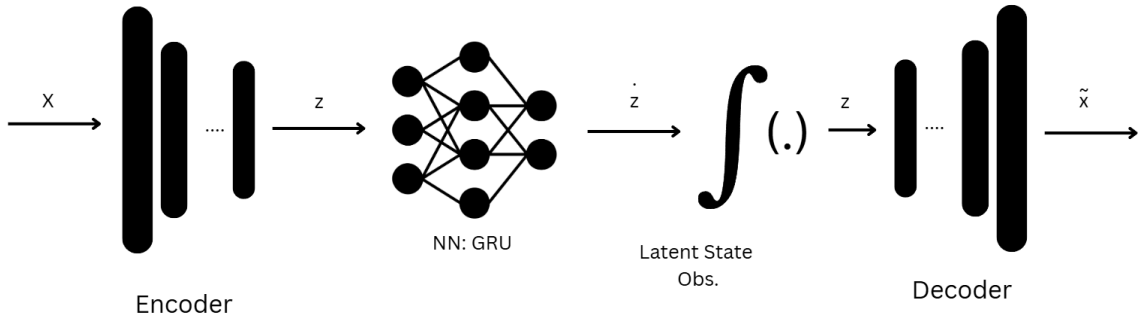
In chaotic systems, SSMs are particularly useful due to their ability to:

1. **Generalize to unseen initial conditions:** The latent state space captures the global structure of system dynamics, enabling predictions for states not present in the training set.

2. **Incorporate mechanistic priors or constraints:** Physics-informed SSMs can embed domain knowledge, improving long-term stability and interpretability.

3. **Support multiple prediction tasks:**
   Filtering: Estimating the current latent state given past and current observations.
   - **Smoothing:** Estimating past latent states using the full observation sequence.
   - **Forecasting:** Rolling out the dynamics to predict future states and projecting them onto observed variables.

These capabilities make nonlinear SSMs an attractive alternative to naive autoregressive methods for predicting chaotic trajectories, offering both robustness and theoretical grounding in control and dynamical systems theory.



2.3 Neural Networks for Chaotic Systems

Neural networks (NNs) are a fundamental tool for modeling high-dimensional, nonlinear, and chaotic time-series data due to their universal function approximation capabilities. At their core, NNs consist of layers of interconnected nodes (neurons), where each node performs a weighted sum of its inputs followed by a nonlinear activation function such as tanhf_0\tanhtanh, ReLU, or sigmoid.

2.3.1 Feed-Forward Networks

Feed-forward neural networks (FFNNs), or multilayer perceptrons (MLPs), propagate information in one direction—from input to hidden layers and finally to output—without recurrent connections. Each hidden layer transforms the input features nonlinearly, enabling the network to approximate complex functions and capture high-order relationships. While FFNNs are effective for regression and independent data tasks, they lack temporal memory, making them less suitable for sequential prediction in chaotic systems.

2.3.2 Recurrent Neural Networks

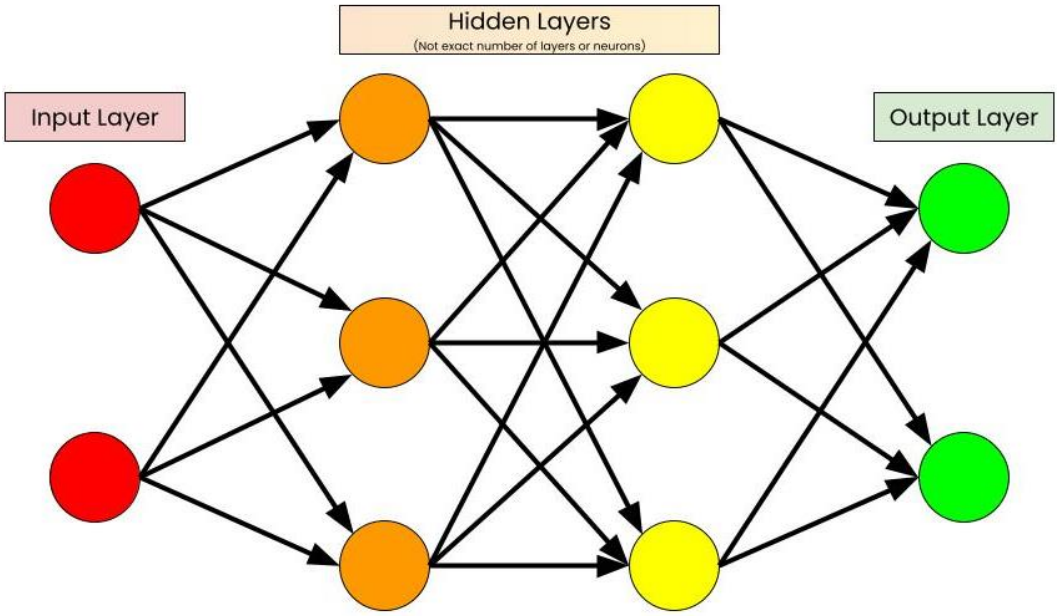
Recurrent neural networks (RNNs) extend FFNNs by incorporating cyclical connections that maintain a hidden state, allowing the network to retain information across time steps. Each neuron receives both the current input and its previous hidden state, enabling the network to model temporal dependencies. RNNs are therefore suitable for time-series forecasting and sequence modeling, including chaotic trajectories.

However, vanilla RNNs suffer from vanishing and exploding gradient problems, which degrade performance for long sequences. To address these issues, gated architectures such as Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) networks are commonly used:

- LSTM: Introduces input, output, and forget gates, along with a separate long-term memory cell, enabling retention of information across long sequences.

- GRU: Simplifies LSTM architecture by combining input and forget gates into a single update gate and eliminating the separate memory cell, offering computational efficiency while maintaining long-term memory capabilities.

These architectures allow neural networks to model the complex temporal dependencies of chaotic systems like the Mackey-Glass equation effectively, providing a strong benchmark for evaluating the proposed time-step-based operator learning framework.



2.4 Time-Step Based Operator Learning

Traditional sequence modeling methods for chaotic systems, including autoregressive recurrent networks and sliding-window forecasts, treat trajectory prediction as a sequential task: each predicted state depends on previous observations or predictions. In systems exhibiting chaos, characterized by positive Lyapunov exponents, even minute errors compound exponentially over time, severely limiting long-term prediction accuracy. Autoregressive RNNs, LSTMs, or GRUs inherently suffer from this compounding error, making them unsuitable for accurate forecasting over long horizons without frequent ground-truth updates.

2.4.1 Classical Operator Learning

Operator learning provides an alternative paradigm. Classical neural operators, such as DeepONets or Fourier Neural Operators, learn mappings between function spaces, for example:

$G: f \mapsto u(\cdot)$

where u(x) denotes a function over a spatial domain and G[f](x) represents its evolution under the governing dynamics, e.g., the solution of a PDE at a fixed time. These architectures are optimized for high-dimensional or infinite-dimensional systems, enabling mesh-independent approximations and systematic generalization across spatial configurations.

In contrast, for low-dimensional chaotic systems such as the scalar Mackey–Glass delay differential equation, deploying full operator architectures is computationally intensive and often unnecessary. The network must encode high-dimensional functional representations and learn

trajectory-level generalization, despite the underlying dynamics being low-dimensional. Consequently, while neural operators are theoretically capable of modeling such systems, their computational overhead outweighs practical benefit. This motivates the adoption of more targeted architectures that efficiently capture cross-trajectory dynamics without incurring the cost of unnecessary functional complexity.

2.4.2 Time-Step Operator Framework

To address these limitations, we introduce a time-step-based operator learning framework. Instead of taking the entire trajectory as input, our operator maps each initial condition and time step directly to the system state:

St: $(x0,t) \mapsto x(t)$,

where x0 is the initial condition and t is the normalized time. Conceptually, the time-based operator represents a pointwise functional map over the trajectory space, which, when trained across an ensemble of initial conditions, learns the global dynamics of the chaotic attractor.

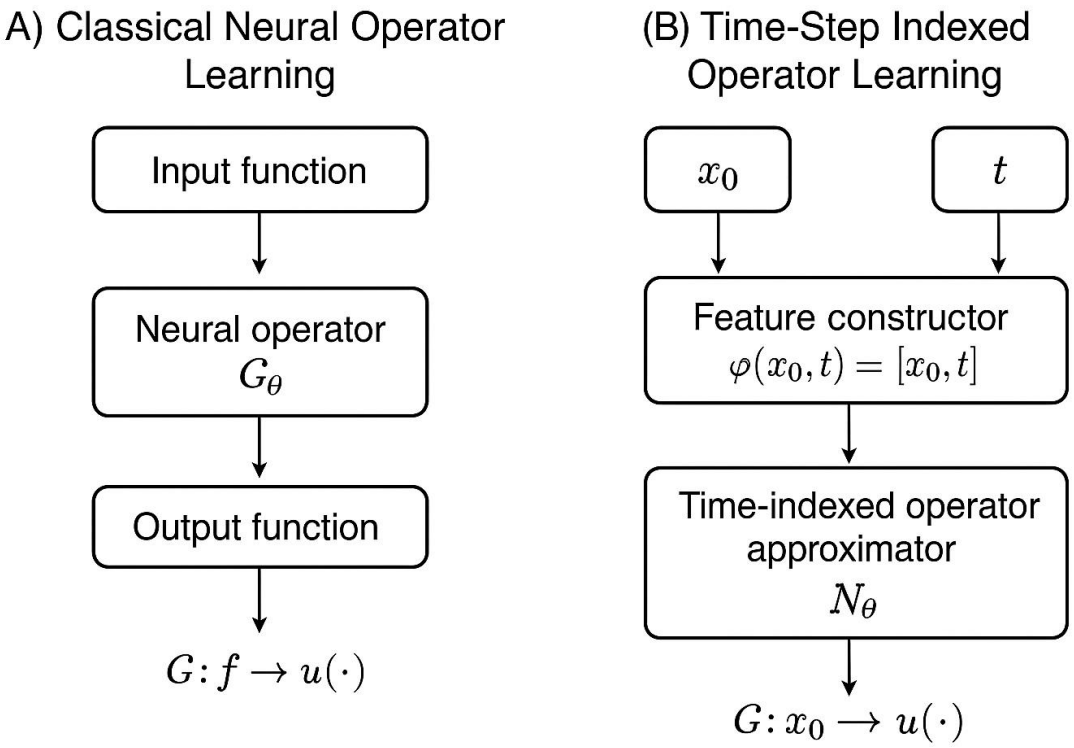


Figure X. *Comparison between traditional operator-learning architectures and the proposed time-step-indexed operator-learning framework.*
(A) In classical neural operators such as the Fourier Neural Operator (FNO) or Deep Operator Network (DeepONet), the model approximates a global operator $G:f \mapsto u(\cdot)$, mapping an entire input function—typically a forcing function, coefficient field, or boundary condition—to a full output function on the solution manifold. These methods rely on global functional representations using lifting layers, Fourier kernels, or branch–trunk decompositions to learn operator-valued mappings.
(B) In contrast, the proposed architecture learns a time-indexed operator $G:x0 \mapsto u(\cdot)$, where the operator domain is restricted to initial conditions instead of full input functions. The model takes the initial state x0x_0x0 and a continuous or discretized time-index ttt, and evaluates the solution generatively as a function of both. This formulation converts operator learning into a query-based functional evaluation problem: the operator produces the value of the solution at any desired time step, without learning a global spatiotemporal kernel. The two mappings differ fundamentally in both the domain of the operator and the computational mechanism by which the function u(t) is reconstructed.

Traditional neural operator frameworks are designed around the assumption that the operator takes entire functions as inputs. Methods such as FNO, DeepONet, and graph kernel operators learn global mappings over function spaces by processing discretized functional inputs through spectral transforms, integral kernels, or branch–trunk networks. This structure inherently ties the operator to the resolution and structure of the input function, and it requires large, diverse datasets of complete function-to-function pairs. In contrast, our method redefines the operator's domain: rather than learning a mapping from functional inputs to functional outputs, we learn a time-indexed operator whose input is only the initial condition. By conditioning the model on both x0 and a time query t, the operator becomes a pointwise generator capable of returning u(t) at arbitrary temporal locations. This formulation eliminates the need to learn a global convolutional or spectral kernel, removes discretization dependence, and directly adapts to chaotic systems whose dynamics cannot be reliably modeled by a single global functional mapping. Unlike traditional operators, which attempt to learn the entire solution manifold in a single transformation, our operator evaluates the manifold locally in time while still preserving global consistency across the trajectory. This represents a conceptual redefinition of operator learning tailored to systems where the governing dynamics are sensitive to initial conditions, highly nonlinear, or chaotic.

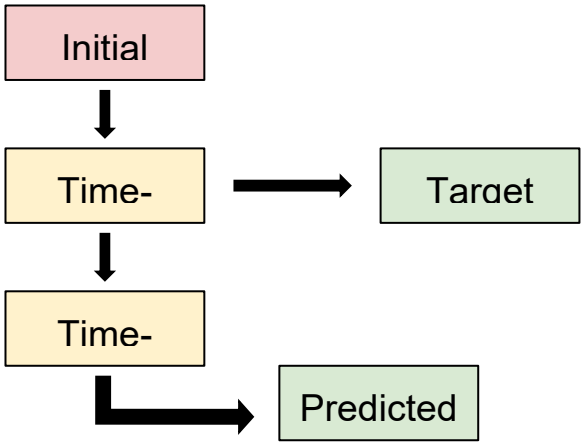


Figure X.

2.4.3 Training Input–Output Structure

Concretely, each training instance consists of a two-dimensional input vector at every time-step in the form of:

$$U_t = [x_0, t/t_{max}]$$

Where x_0 is the initial condition and t/t_{max} is the normalized time index. For a given trajectory of length T, these inputs are stacked to form a matrix of shape (T, 2), representing the system's progression from the same initial condition over time. The corresponding target vector is a vector x(t) $\in \mathbb{R}^T$ representing the true temporal evolution of the system, obtained via numerical integration through Euler's method as outlined in Section 2.0.

Input: $[(x_0,t/T)_0 , (x_0,t/T)_1 ,\ldots, (x_0,t/T)_{T-1}] \rightarrow$ Target: $[x(0), x(1) ,\ldots, x(T-1)]$

Unlike traditional time-conditioned regression, our model is trained over an ensemble of trajectories with distinct initial conditions, forcing it to learn the global structure of the system's

flow map rather than memorizing a single trajectory. This distinction fundamentally differentiates our approach from standard data-driven regressors, enabling cross-trajectory generalization—a property central to chaotic dynamics modeling.

Critically, one initial condition is held out entirely from training ($x_0 = 1.2$). The model is tested on its ability to reconstruct the full trajectory given only the held-out initial condition and the normalized time indices, with no access to the actual system evolution. This held-out evaluation is especially relevant in clinical or biological modeling scenarios, where only a small set of observed patient or system trajectories may be available for training. The ability to accurately predict the time-evolution of an unobserved instance from initial parameters alone reflects the model's robustness and generalizability—key traits for applications such as personalized medicine, where individual dynamics may deviate from the observed population but must still be inferable based on initial state features.

Unlike traditional time-conditioned regression, our model is trained over an ensemble of trajectories with distinct initial conditions, forcing it to learn the global structure of the system's flow map rather than memorizing a single trajectory. This distinction fundamentally differentiates our approach from standard data-driven regressors, enabling cross-trajectory generalization—a property central to chaotic dynamics modeling.

2.5 Exclusion of Transformer-Based Architectures and ARIMA Networks

While Transformer-based architectures have demonstrated state-of-the-art performance in sequential modeling tasks, they were excluded from this study due to their incompatibility with our time-based neural operator. The proposed framework learns a direct functional mapping between the initial condition and normalized time step $(x0,t) \rightarrow x(t)$ without relying on autoregressive context or historical data. In contrast, transformers depend fundamentally on self-attention mechanisms across discrete input sequences, where positional encodings define relationships among tokens. When presented with non-sequential, continuous-valued inputs, the attention operation collapses, providing no meaningful temporal dependencies to exploit.

Furthermore, the continuous and low-dimensional structure of our input space renders standard positional encodings ineffective and leads to training instability. Even advanced adaptations, such as continuous-time Transformers or neural-ODE variants, would require substantially larger

datasets and computational budgets, introducing overparameterization without performance gains.

Other traditional sequence models—including autoregressive (ARIMA) and sliding-window recurrent networks—were similarly omitted, as they depend on recursive forecasting from past samples. Our framework instead focuses on learning the system's flow map, generalizing across initial conditions rather than predicting sequentially from prior states. Therefore, the architectures evaluated in this work—Reservoir Computing, Gated Recurrent Units (GRUs), Long Short-Term Memory (LSTM) networks, and nonlinear State-Space Models—were selected for their compatibility with continuous-time mappings and their capacity to generalize to unseen trajectories within chaotic dynamical regimes.

2.7 Comparison Metrics

To evaluate the models, we implemented the following comparison metrics: R^2, RMSE, F-statistic, Pearson correlation, spectral coefficient similarity, and, most importantly, divergence time.

Pearson: The Pearson correlation coefficient (r) measures the linear relationship between predicted and observed values, with values ranging from −1 to 1. A value of 1 indicates perfect positive linear correlation, −1 indicates perfect negative correlation, and 0 indicates no linear correlation. In the context of neural networks, r provides a standardized metric to assess how well a model captures the trends in the data. Similar to the R^2 values, Reservoir computing had the highest |r| of 0.9885, with FFNN having the lowest |r| of 0.8090.

RMSE: The Root Mean Squared Error (RMSE) quantifies the average magnitude of the errors between predicted and observed values by calculating the square root of the average squared differences. It provides a measure of how well the model's predictions align with the actual data, with lower values indicating better predictive accuracy. RMSE is particularly useful when large errors are undesirable, as it gives greater weight to larger deviations. In neural network studies, RMSE offers a straightforward metric to compare the overall error across different models. For the Mackey-Glass system,

R^2: The R^2 (coefficient of restitution) measures the proportion of variance in the observed data that is explained by the model's prediction. A value of 1 indicates the model performing perfectly, whereas 0 represents the model explaining none of the variance. R^2 is particularly useful in assessing how well a model captures the overall trends, complementing metrics like RMSE and Pearson correlation by providing a normalized measure of predictive performance. For the Mackey-Glass system,

F-Statistic: The F-statistic is a ratio that compares the variance explained by a model to the variance explained by the model. It is used to assess the overall significance of a model by testing whether the model's predictions are significantly better than predicting the mean. Higher values of the F-statistic indicate that the model explains a greater proportion of the variance, suggesting that it is a better fit for the data. In the context of neural network studies, the F-statistic is often used to compare the relative performance of different models. For the Mackey-Glass system, Reservoir Computing yielded the highest F-statistic of 4047.7212, demonstrating its superior model fit compared to Feed Forward, which had the lowest F-statistic of 147.8296.

Spearman coefficient: The Spearman rank correlation coefficient measures the monotonic relationships between predicted and observed values, assessing how well the predictions are able to gauge the ordering of the data. Values range from 1 to -1, where 1 indicates a perfect increasing rank correlation, -1 represents a perfectly decreasing rank correlation, and 0 indicates no monotic correlation. Spearman correlation is particularly useful when the relative trends are important, specifically in timeseries based data where the current data point builds on the previous. \rho is able to provide insight into how well the model captures the overall ranking of the outputs. For the Mackey-Glass system, Reservoir Computing achieved the highest Spearman coefficient of 0.9852, showing the strongest preservation of the data's order, while GRU had the lowest Spearman coefficient of 0.6993.

Spectral cosine similarity: Spectral cosine similarity measures the similarity between two signals or vectors based on the angle between them in a high-dimensional feature space. Values range from -1 to 1, with 1 indicating identical direction (maximum similarity), 0 indicating orthogonality (no similarity), and -1 indicating opposite direction. In the context of neural networks predicting time series,

it provides a metric for comparing the shape or pattern of predicted trajectories against observed data, independent of magnitude. Reservoir Computing had the highest spectral cosine similarity of 0.9992, while FFNN had 0.8350.

Divergence Time: Divergence time quantifies the difference between predicted and observed trajectories by measuring the temporal point at which they deviate beyond a specified threshold. In this study, we define this threshold as an error margin of 0.05, meaning divergence is recorded when the absolute difference between predicted and true values exceeds 0.05. This metric provides a measure of how long a model can reliably forecast the system before predictions become significantly inaccurate.

RESULTS

To evaluate the timestep-based framework, we implemented and compared multiple machine learning architectures, including feedforward Neural Networks (NNs), State Space Computing (SSC), and Reservoir Computing (RC). Model performance was first assessed across a structured set of initial conditions, with x0 sampled over two subranges, 0–1.17 and 1.23–3.0, with 15 initial conditions in each, yielding a total of 30 distinct states for analysis.

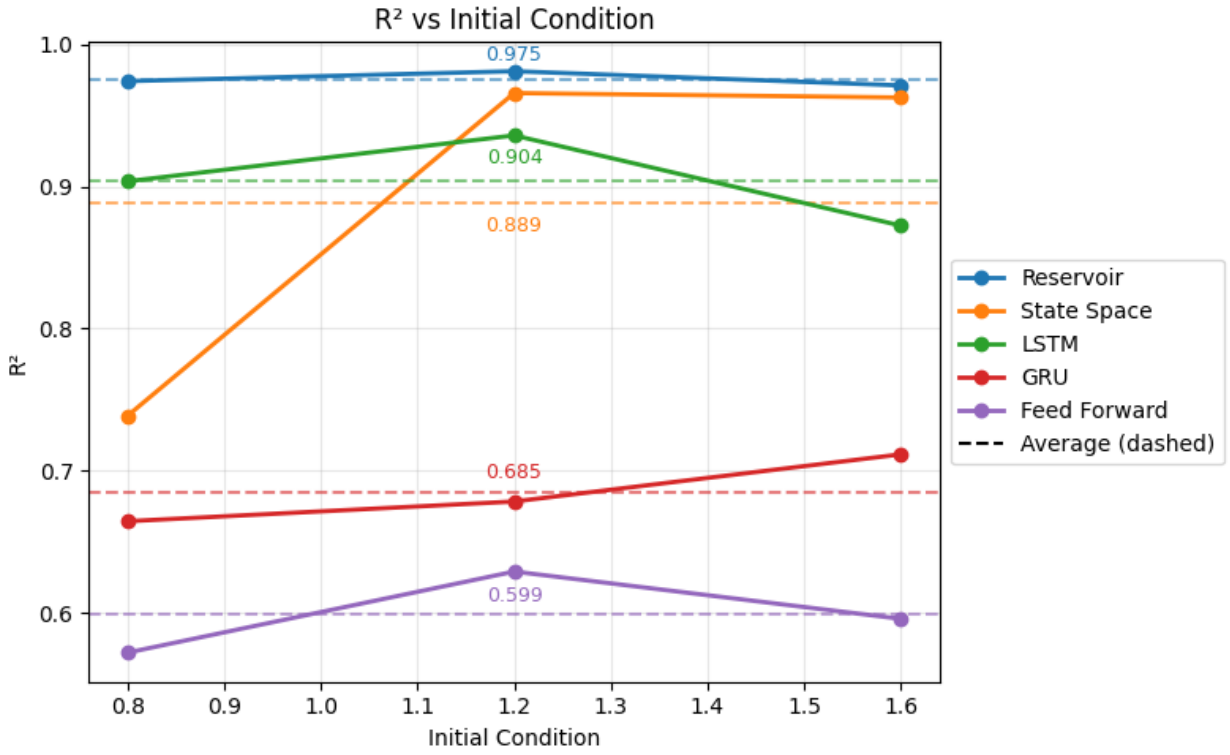
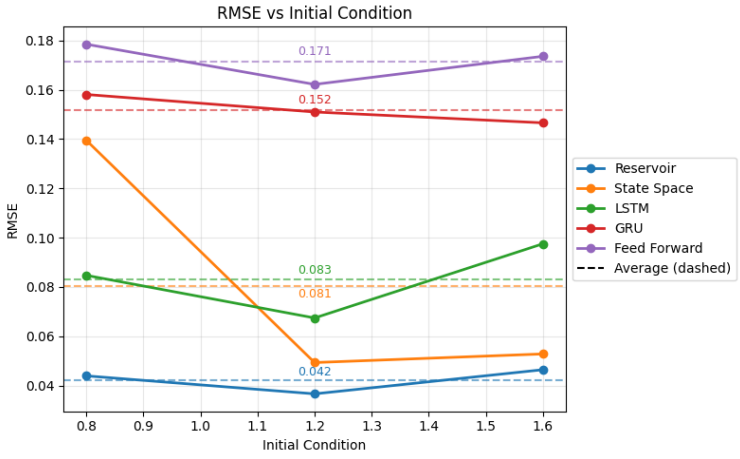


Figure X. *R^2 scores of all tested models over varying initial conditions.*

Reservoir Computing achieved the highest R^2 of 0.9754, indicating it captured most of the variance in the data, while Feed Forward had the lowest R^2 of 0.5988, reflecting comparatively weaker explanatory power.

RMSE:



Reservoir Computing achieved the lowest RMSE of 0.0423, indicating the smallest average error in predictions, while Feed Forward had the highest RMSE of 0.1714.

Figure X. *RMSE scores of all tested models over varying initial conditions.*

Divergence Time:

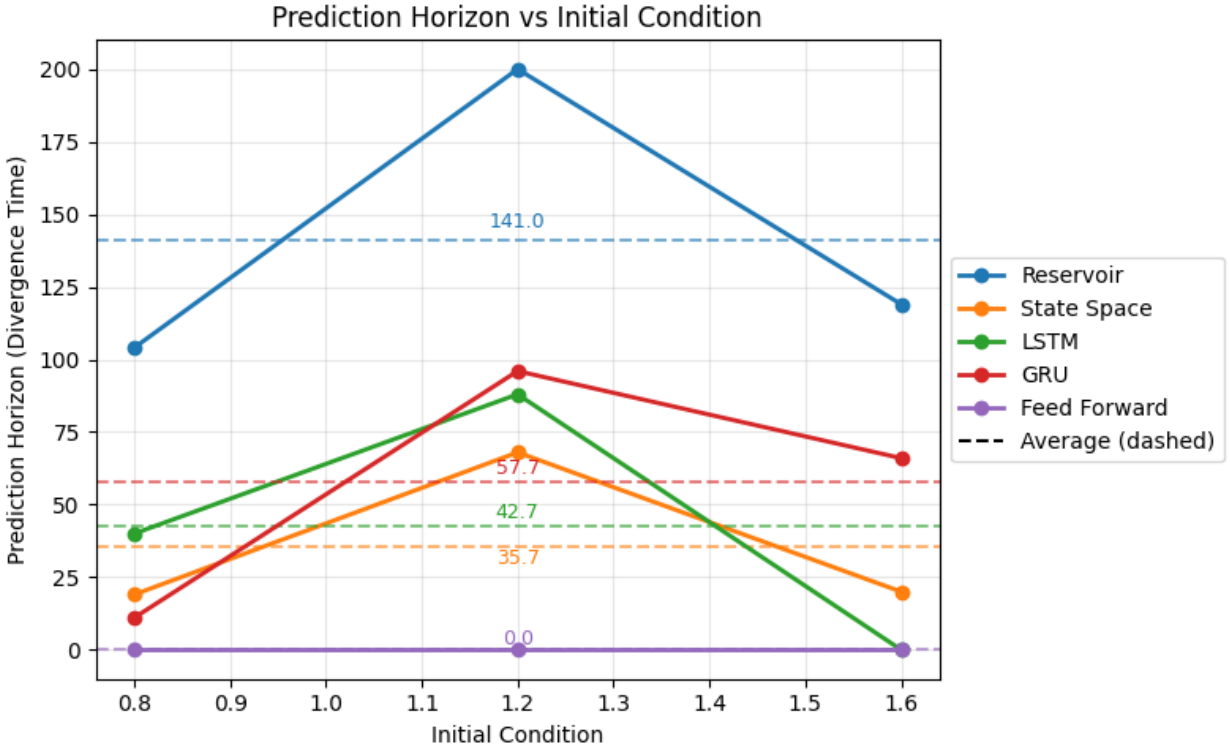


Figure X. *Divergence time of all tested models over varying initial conditions.*

While Reservoir Computing had a divergence time of 141 timesteps, the error was often less than 0.05 for most of the 200 time steps, showcasing how the model can easily capture short to long-term dynamics without a full divergence. On the other hand, FFNN had a very short divergence time of 0, and eventually did fully diverge from the actual trajectory.

Baseline:

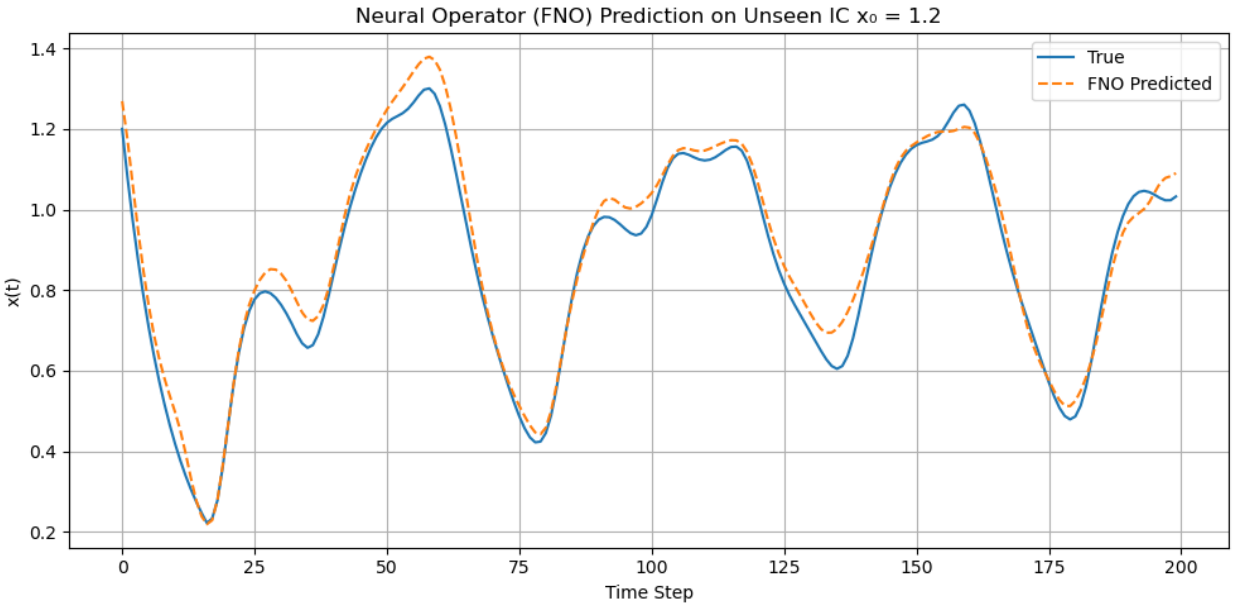


Figure X. *FNO Trajectory with initial condition 1.2*

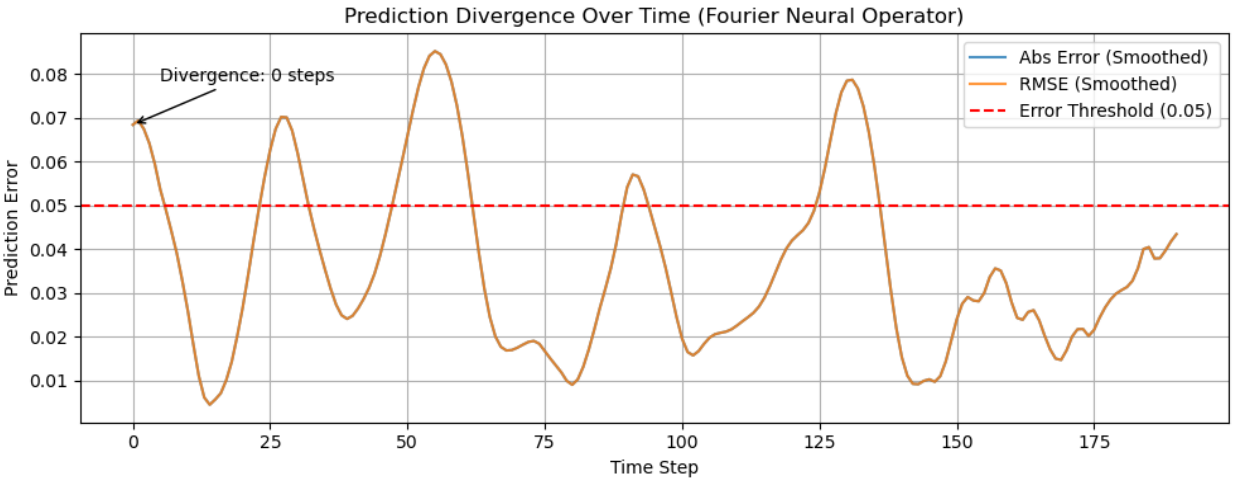
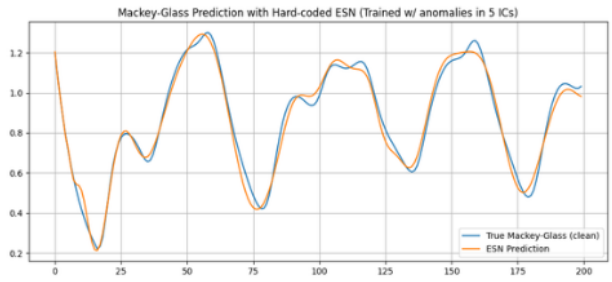
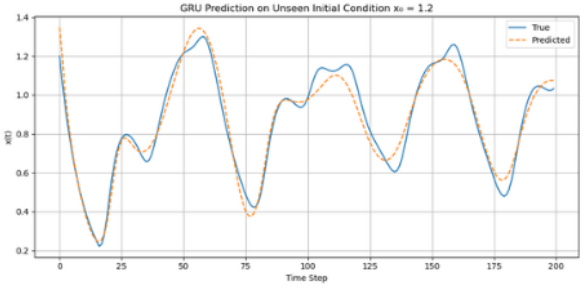


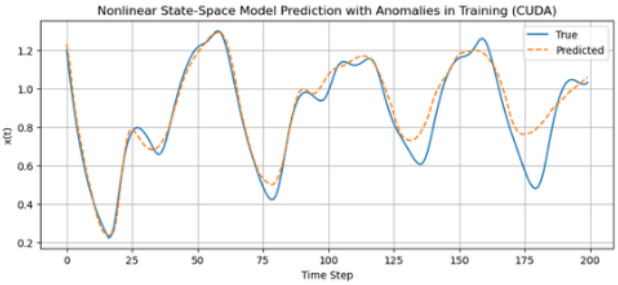
Figure X. *Divergence graph of FNO. It diverges at 0 timesteps.*

US:

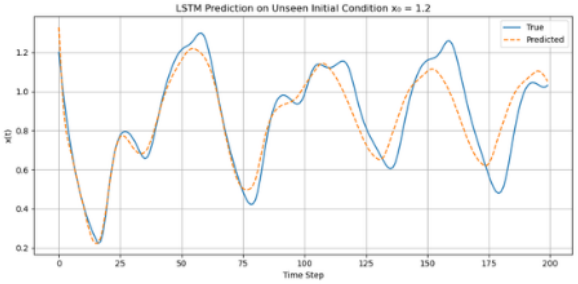




B)

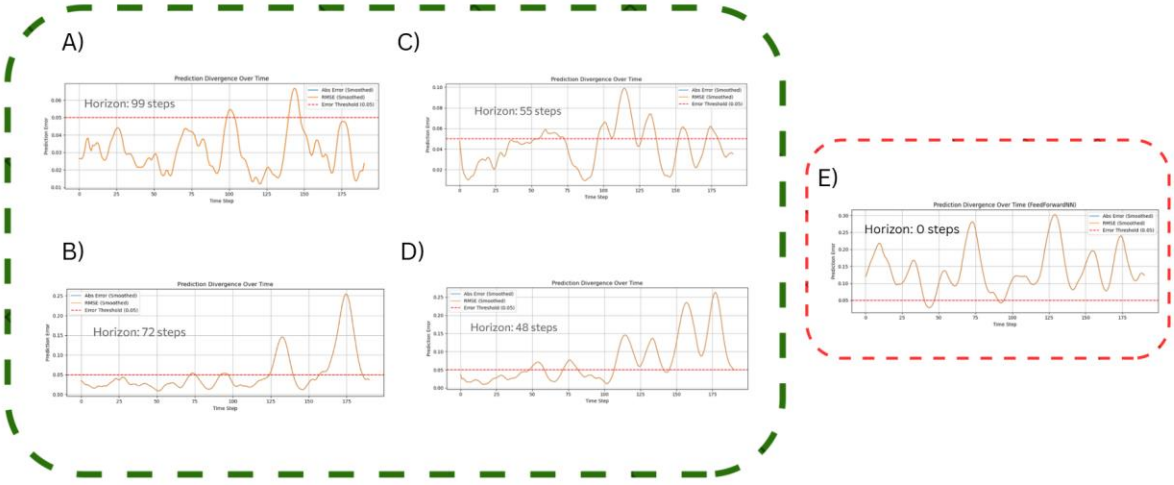


D)



Comparison of Mackey–Glass Predictions from Five Models | Time-Based Neural Operators

Better than FNO (A-D), Worse than FNO (E)



3. Results and Discussion

3.1: Limitations

While the proposed framework demonstrates strong performance on the Mackey–Glass system, several limitations should be acknowledged. First, the study employs a synthetic delay-differential equation under noise-free conditions, which may not fully capture the stochasticity present in biological systems. Nonetheless, the Mackey–Glass model remains a canonical benchmark for evaluating chaotic prediction methods. Second, the anomaly injection process—implemented as controlled perturbations to add noise to the system—was designed to assess robustness in more chaotic scenarios rather than simulate real-world measurement artifacts. Third, the framework was applied to a single system; extending it to other nonlinear dynamics (e.g., Lorenz, Rössler, or real biomedical signals) constitutes an important next step. Finally, although our focus was on predictive capability, future work may explore model interpretability and connections between the learned mappings and physical state-space structure.

3.2: Discussion

The results of this study establish Reservoir Computing as a highly effective framework for reconstructing the chaotic behavior of the Mackey–Glass system. Moreover, introducing stochastic perturbations during training improves stability and promotes stronger out-of-sample performance under perturbed initial conditions. Relative to conventional feedforward neural networks and state-space modeling approaches, the reservoir-based architecture achieves comparable or superior predictive accuracy while requiring substantially less computational overhead.

While other studies have used methods such as the FNO or autoregressive approaches, these observations suggest that the timestep methodology developed here is well suited for extension to complex physical and engineered systems in which nonlinear feedback and stochastic disturbances interact to produce apparent unpredictability. In contrast, within our approach, the FNO struggles to generalize in chaotic delay systems when trained as a static operator over sparsely sampled initial conditions.

Specifically, the approach avoids constraining the model to a predetermined window of past observations, allowing the reservoir's internal state to evolve continuously. This implicitly encodes extended temporal context without requiring explicit lag selection or manual feature engineering, thereby providing greater flexibility, greater stability under perturbations, and improved generalization when modeling systems characterized by sensitive dependence on initial conditions. By moving beyond sliding-window constraints,

the proposed approach offers a scalable framework for forecasting and monitoring high-dimensional dynamical systems in practical settings. Potential domains of application include atmospheric and climate dynamics, financial market forecasting, nonlinear control systems, biological regulatory processes, and chaos-based secure communication schemes.

4. Conclusion

The timestep-based Reservoir Computing framework offers more than accurate trajectory reconstruction; it provides a window into the structure of chaotic dynamics in the Mackey–Glass system. By evolving high-dimensional internal states across sequential timesteps, the reservoir implicitly captures delayed feedback and long-range correlations that are inaccessible to linear or fixed-window approaches. Analysis of Lyapunov exponents on predicted trajectories confirms that the method preserves the system's inherent exponential divergence, allowing us to quantify the chaotic sensitivity of different regions of the state space. Introducing stochastic perturbations further highlights which initial conditions are most prone to divergence, effectively mapping the system's stability landscape and identifying potential tipping points.

These insights have direct implications for forecasting and control in complex systems with time delays. In physiological modeling, for example, small deviations in initial conditions can lead to drastically different outcomes in blood cell populations or hormone cycles, and the reservoir approach can help identify thresholds for intervention. In engineered systems with delayed feedback, such as robotic control loops or signal-processing networks, the method can anticipate instabilities before they propagate, enabling the design of more robust controllers. Moreover, the high-dimensional embeddings created by the reservoir make the system particularly well-suited for anomaly detection in chaotic environments, offering a means to flag unusual or potentially hazardous behavior in real time.