Towards Modeling Physically-Consistent, Chaotic Spatiotemporal Dynamics with Echo State Networks

Matthew Ziemann[*†], **Alisha Sharma**[*‡], **Kaiyan Shi**[*], **and Yiling Qiao**[*]

[*] University of Maryland, College Park, MD 20742
[†] Army Research Laboratory, Adelphi, MD 20783
[‡] Naval Research Laboratory, Washington, DC 20375
{mrziema2,ajsharma,kshi12,yilingq}@umd.edu

Abstract

This study explores how echo state networks (ESNs) can be used in time-series forecasting of chaotic physics. We compare the performance of a basic ESN with two physics-informed variants, tested on the canonical Lorenz attractor. We then apply the ESN to a large-scale atmospheric model and a larger real-world weather dataset to test its ability to scale to large spatiotemporal systems. We find that a traditional ESN when properly tuned can outperform our equivalent physics-informed methods. We also find that the ESN is capable of accurately predicting the global evolution of the atmospheric primitive equations over short time frames (∼67 hrs), but struggles to accurately predict real-world data.

Introduction

Many useful scientific simulations, such as large-scale climate models (Fig. 1), contain computationally intractable subroutines that limit the pace and accessibility of research in critical areas. Neural networks are gaining popularity as a way of sidestepping these bottlenecks due to their reduced computational complexity, improved scalability, and low post-training cost (Frank, Drikakis, and Charissis 2020). However, they also have some significant disadvantages that prevent widespread adoption. Notably, they require massive amounts of data and computational resources to train. Furthermore, without explicit knowledge of physics, their predictions can be unreliable and break important physical laws, which can be disastrous in scientific simulations. This is particularly true of the difficult regimes that engineers often want to bypass, such as chaotic, stiff, or multiscale systems.

Echo state networks (ESNs), a simplification of recurrent neural networks, address several of these challenges. ESNs are one of the most effective data-driven architectures for time-series forecasting, particularly for chaotic dynamical systems (Aggarwal 2018; Jaeger 2001). While they share the temporal invariance of standard recurrent neural networks, all parameters aside from the final fully-connected layer are fixed, making them extremely cheap to train. Unfortunately, while ESNs excel at modeling low dimensional dynamics, they scale poorly to high-dimensional input (Aggarwal 2018), which poses a challenge for problems with many degrees of freedom such as those with spatial parameters. Furthermore, while they can accurately model complex chaotic systems, they are purely data-driven: they have no qualms about breaking the laws of physics.

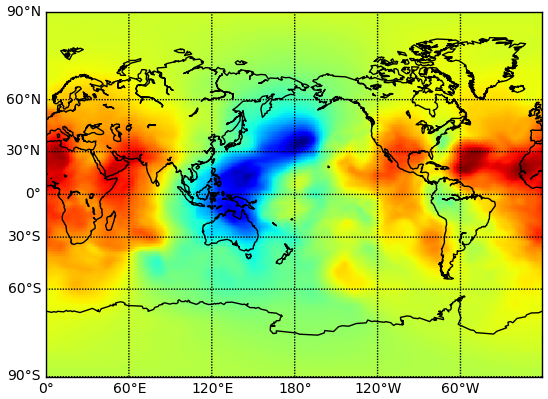

Figure 1: Velocity potential computed using the Climate Forecast System (CFS) (Saha et al. 2010, 2014).

Researchers have suggested various methods to overcome these challenges. Studies suggest that informing neural networks with physics-based knowledge can yield reduced requirements for data and training time, as well as significantly improving overall performance. Two examples of these methods include the "Combined Hybrid/Parallel Prediction" (CHyPP) method (Wikner et al. 2020), and physics-informed loss constraints (Raissi, Perdikaris, and Karniadakis 2019; Doan, Polifke, and Magri 2019). We present a comparative study of these techniques to better understand their respective benefits and limitations on the small-scale chaotic systems and extended these methods to explore their feasibility in large-scale dynamical systems.

The first two authors contributed equally to this work.

Echo State Networks

Recurrent neural networks (RNNs) are a natural model for dynamical systems, but they can be difficult to train in practice. Echo state networks (ESNs) are a promising simplification of RNNs: they freeze most of the parameters, swapping a highly non-convex training process with a simple convex regression, but still retain much of the sequential structural and expressive power of general RNNs (Jaeger 2001).

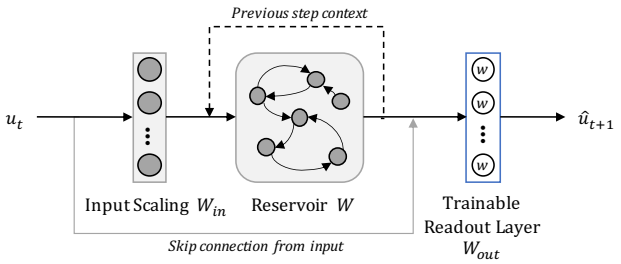

Figure 2: Diagram of a basic echo state network (ESN). The readout layer is trained by ridge regression to map the nonlinear reservoir projections to the target output state.

Conceptually, ESNs work similarly to other weighted average methods: they use the architecture of a general RNN to (a) generate high-dimensional nonlinear projections of the inputs and (b) learn an optimal weighted average of these projections to predict the next state (Figure 2). The input scaling and reservoir layers are initialized according to the "echo state property" (Jaeger 2001) and then are frozen. Learning the readout layer is a convex minimization problem between the reservoir outputs and the output state; the global minimum can be found cheaply through closed-form ridge regression or by using a convex solver (Jaeger 2001).

Physics Informed Echo State Networks

We focus on two mechanisms for embedding physics into the basic ESN architecture described above: a *physical consistency loss function* and a *hybrid data-driven/knowledge-based predictor*. These approaches are described in the following sections.

Physical Consistency Loss

A direct way to constrain the network is by explicitly embedding the system's governing dynamics into the optimization of the readout layer. Through training, the ESN should learn to emulate the system's governing dynamics.

Raissi, Perdikaris, and Karniadakis (2019) introduced an interesting discrete mechanism for this using Runge-Kutta solvers, a family of iterative algorithms for numerically solving systems of ordinary differential equations. Given an initial value and set of governing equations, Runge-Kutta solvers estimate the next timestep by taking a weighted average of several intermediate predictions called stages. This formulation fits naturally with the discrete nature of ESNs.

If we consider the ESN to be a surrogate Runge-Kutta solver, we can verify its prediction by applying the Runge-Kutta equations in reverse at each stage. If the ESN is a good surrogate, this "round trip" will end back at the initial state. Training minimizes the distance between the true and calculated initial states, providing a strong error signal and bringing the ESN predictions closer to the solver.

This physical consistency term has a side effect of making training harder, turning a convex problem highly nonconvex.

Hybrid Prediction Model

As many systems of interest have known reduced or approximate formulations, an alternate strategy for improving physical consistency is based on the idea that correcting similar (but incorrect) dynamics is easier than learning a system from scratch. By concatenating steps of a reduced-order or flawed model of the physical system to the reservoir output, we can reformulate the learning problem as *correcting* the flawed dynamics instead of predicting them from scratch (Wikner et al. 2020). As in the basic ESN, training is convex.

Preliminary Results

Three echo state network (ESN) configurations were tested: a basic ESN (**BaseESN**), an ESN trained with a physical consistency loss (**PhyESN**), and a hybrid numerical/data-driven approach where BaseESN was used to refine a reduced-order estimate (**HyESN**). Models were implemented in Julia using the SciML ecosystem (Martinuzzi 2020).

We first trained our three models to predict the spatiotemporal evolution of a simple, well-studied chaotic attractor: the Lorenz attractor. We then implemented the best performing ESN with two large-scale dynamical systems of increasing complexity to evaluate its ability to scale to more difficult problem sets.

The models were evaluated by two metrics. First, their **relative accuracy** was measured by the time-averaged root mean squared error (RMSE) over a fixed prediction time. Next, the **dynamic stability** of a model was measured by the time horizon, or time for which the normalized error of the model predictions stays under a specified error tolerance ϵ_{max} (Wikner et al. 2020; Doan, Polifke, and Magri 2019).

Lorenz Attractor

The first set of experiments focused on the chaotic Lorenz attractor, a canonical problem in chaotic dynamics modeling. The Lorenz attractor is the chaotic regime of a dynamical system derived from an atmospheric surface convection model (Lorenz 1963). The governing equations are given below:

$$\frac{\partial u}{\partial t} = \begin{bmatrix} \sigma(u_2 - u_1) \\ u_1(\rho - u_3) - u_2 \\ u_1 u_2 - \beta u_3 \end{bmatrix} \tag{1}$$

System behavior is controlled by the parameters σ, ρ, and β, and the most commonly chosen parameters to study chaotic dynamics are $\sigma = 10, \rho = 28$ and $\beta = 8/3$.

Numerical results are summarized in Table 1. Results are calculated for the best models. Each hyperparameter configuration was tested over several thousand random initializations for BaseESN and HyESN; in contrast, PhyESN was only tested over 5 initializations per configuration due to the high training cost. Results are reported in Lyapunov times, the characteristic timescale of a chaotic system, as in (Wikner et al. 2020; Doan, Polifke, and Magri 2019).

BaseESN In the first experiment, we modeled the Lorenz system with a basic ESN (BaseESN). The basic ESN architecture includes an input layer initialized with a random uniform distribution with values in $[-\sigma, \sigma]$, a random sparse reservoir with average connectivity d and spectral radius α, and a T_1 nonlinear transformation of the reservoir output (Chattopadhyay, Hassanzadeh, and Subramanian 2020). The readout layer was trained using ridge regression with regularization weight β. A skip connection was included from the input to reservoir output, and the reservoir leaked previous inputs to the readout layer with leak coefficient a.

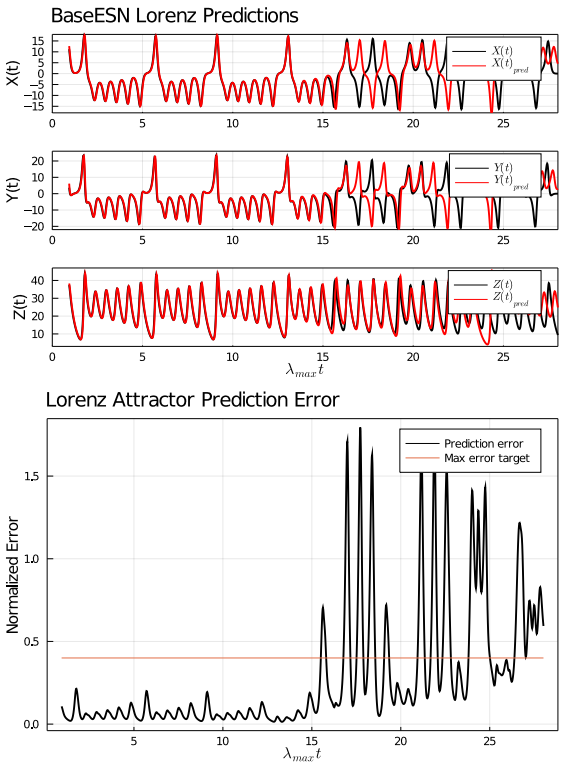

Figure 3: BaseESN Lorenz Predictions and Error

The BaseESN predictions diverged from the true system after approximately 15 Lyapunov times, though the predictions continued to follow a similar chaotic pattern (Figure 3). BaseESN was also extremely fast to train and predict: training the network using closed-form ridge regression and predicting 1000 new points (approximately 20 Lyapunov times) each took ≈ 50 ms (Table 1). We also measured training time with a BFGS solver, an iterative nonlinear minimizer, to provide a performance baseline for PhyESN; this approach was nearly 4 orders of magnitude slower (≈ 3 s).

PhyESN The second experiment (PhyESN) took the same baseline ESN architecture and added a physical consistency loss. This physical consistency loss function was significantly more difficult to optimize than the ridge regression in BaseESN: not only is it non-linear, it is non-convex. To account for this, the readout layer was trained using BFGS.

Qualitatively, the PhyESN results looked similar to BaseESN. However, despite the physics embedded in the loss function, the best PhyESN models had slightly worse accuracy and stability than the BaseESN. Furthermore, training took significantly more time: BFGS struggled to converge, taking 5-7 orders-of-magnitude longer (depending on the run) than the closed-form BaseESN training, and the final solution was a local (not global) minimum.

HyESN The trained HyESN predictions diverged from the true system after approximately 9 Lyapunov times (Figure 4), which is worse than the results predicted by BaseESN. However, HyESN frequently exhibits an interesting recovery behavior: in Figure 4, the dynamics appear to recover between 11 and 17 Lyapunov times. This can also be seen in the numerical results: while BaseESN has a substantially longer time horizon, the error over 20 Lyapunov times is 12% lower in HyESN.

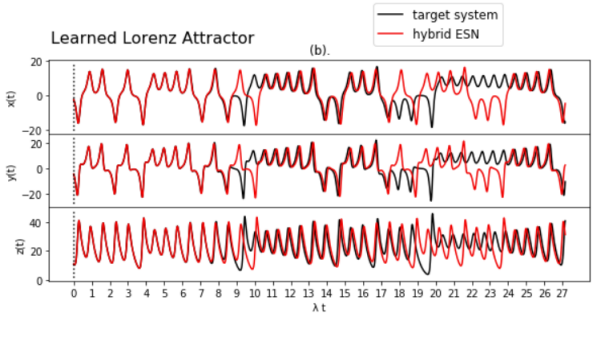

Figure 4: Hybrid Model (HyESN) Lorenz Predictions

Training time for HyESN was consistently slower than BaseESN, though they were similar; while they are both trained with closed-form ridge regression, the HyESN readout layer was larger due to the flawed dynamics input.

Weather Forecasting

We took a two-part approach to evaluate the behavior of ESNs on large-scale systems. First, we trained a BaseESN (our best-performing ESN) on data generated by a simple atmospheric model and evaluate its prediction error. Then

Table 1: Best results, Lorenz Models. The best results are in **bold** font.

Net	RMSE (20 $\lambda_{\max} t$)	Time Horizon ($\lambda_{\max} t$)	Train Time(s)
BaseESN (literature)	1.69	5.10	**1.36e-3 (closed-form)**
BaseESN (tuned)	1.09	**15.13**	**1.08e-3 (closed-form)** 2.98 (BFGS)
HyESN	**8.91e-1**	8.97	5.59 (closed-form)
PhyESN	1.8	7.2	3.0e+2 (BFGS)

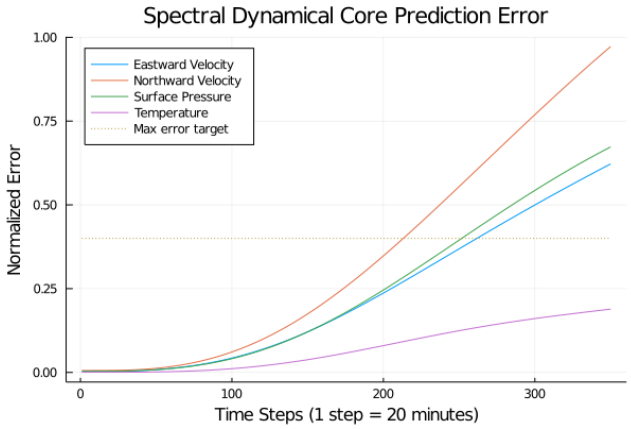

Figure 5: Plot of BaseESN normalized error over time for the primitive equations system.

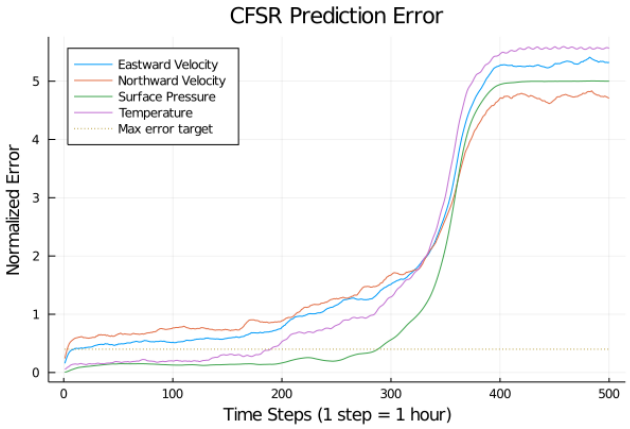

Figure 6: Plot of BaseESN normalized error over time for the CFSR data.

we trained a BaseESN on real-world data from the National Oceanic and Atmospheric Association (NOAA)'s Climate Forecast System (CFS) (Saha et al. 2010, 2014) and evaluated its prediction error.

We begin with the simple atmospheric model. The governing system of nonlinear, partial differential equations for atmospheric dynamics—known as the *primitive equations*—was numerically solved to generate data for pressure (P), temperature (T), and the latitudinal & longitudinal components of wind velocity (u & v) (Ehrendorfer 2011). This system is commonly called the *dynamical core*, as it is the core of most numerical weather models. These four parameters were solved on the surface of Earth with a grid of 64 longitudinal points and 32 latitudinal points. We solved u, v, and T at three altitude levels (surface, mid-, and high-). The pressure was only solved at the surface, bringing the total number of ESN input parameters to 20,480, a significant increase from the 3 input parameters of the Lorenz system. It was trained over 300 days of data with 20 minute timesteps.

With some tuning, the ESN performed quite well in predicting the evolution of states. With our best-performing model, the ESN predicted over 200 timesteps (>67 hrs) while remaining below the max normalized error cutoff threshold (0.4), seen in Figure 5. Notably, the ESN predictions gradually accumulate error over time and do not diverge significantly. When results finally diverge, the ESN continues to predict realistic values, though they do not reflect the actual evolution of the system.

We then trained the BaseESN on real-world data from the CFS Reanalysis (CFSR). We utilized hourly data from January 2005 to December 2006, on a 73x144 spatial grid. Due to memory constraints, we restricted the data to u, v, P, & T rather than utilizing the full 73 parameters of CFSR, and again used three altitude levels for u, v, & T. This increased the number of input parameters to 105,120, approximately 5 times more than required for the dynamical core system.

The BaseESN did not perform as well on the real-world data as it did on the dynamical core system. As seen in Figure 6, the predictions for the velocity components diverged from acceptable error levels within four timesteps, likely because the velocities evolve more quickly. Pressure and temperature both evolve much more slowly over time, and so the BaseESN was able to remain within acceptable error levels for much longer. Despite the quick divergence from real-world values, the BaseESN continued to make realistic (if incorrect) predictions until around 300 timesteps, where it diverged quite suddenly from realistic values. It's worth noting that we were unable to perform significant hyperparameter tuning on the CFSR system as the computational cost was quite high, requiring multiple days to train.

Discussion

We found that a highly tuned BaseESN generally outperformed the physics-informed techniques (PhyESN and HyESN) on the Lorenz system when measured on prediction accuracy and dynamic stability. This result underscores the importance of hyperparameter tuning in deep learning approaches.

The primary advantage to BaseESN is the low cost to find the global minimum in low-dimensionality systems. Hyperparameter selection and reservoir initialization significantly impacted model performance. The stark difference between the BaseESN (baseline) and BaseESN (tuned) results in Table 1 illustrates this well: the tuned BaseESN is stable for approximately $3\times$ longer than the baseline BaseESN model, which uses hyperparameters found commonly in the literature (Wikner et al. 2020; Doan, Polifke, and Magri 2019). The BaseESN's cheap training and prediction cost allow for extensive parameter searches and repeated trials, even when on modest hardware.

PhyESN does not share this benefit. Conceptually, physics constraints were intended to improve the physical consistency (and thus reliability) of the model; however, the practical concerns overshadowed any benefit. PhyESN trains by minimizing the residual during a round-trip ODE solver step. In other words, it takes an easy problem and turns it into a very hard (nonconvex) one. There is no longer a clear closed-form solution, and optimizers are more likely to get stuck in bad local minima. Qualitatively, the loss landscape seemed rough and difficult to traverse: training times for the network varied wildly (though all were many orders of magnitude slower than ridge regression), and first-order gradient descent techniques failed to converge. This technique was a poor match for this experiment, but it may perform better in different network architectures or for systems with different dynamics.

Conceptually, HyESN keeps many of the benefits of BaseESN, including the convex formulation and cheap training cost; however, the HyESN's hybrid approach did not improve the model's time horizon. This may be because the Lorenz system is relatively simple and thus does not benefit from the flawed model's "hints". One interesting feature of HyESN was its recovery ability. As noted previously, HyESN predictions frequently appeared to recover across reservoir initializations, which can be seen in the low long-term prediction error reported in Table 1. If this behavior persists in other systems, this could be a benefit in long-running simulations.

The results from our experiments with the primitive equations were promising and hint toward the ESN's ability to handle high-dimensional inputs better than we anticipated. However, as expected, the computational cost of training ESNs and generating predictions does not scale well with large numbers of inputs, which hurts its feasibility in practice. The use of ESNs for large-scale systems would benefit from research to implement feature reduction techniques—for example, the use of a convolutional autoencoder to encode inputs and decode outputs. Faster training and prediction with ESNs would enable a greater degree of hyperparameter tuning for large systems which may yield stronger results for real-world systems. This would also enable the use of more input data which may lead to further improvements for real-world systems.

References

Aggarwal, C. C. 2018. *Neural Networks and Deep Learning: A Textbook*. Cham: Springer International Publishing. ISBN 978-3-319-94462-3 978-3-319-94463-0. doi:10.1007/978-3-319-94463-0.

Chattopadhyay, A.; Hassanzadeh, P.; and Subramanian, D. 2020. Data-Driven Prediction of a Multi-Scale Lorenz 96 Chaotic System Using Deep Learning Methods: Reservoir Computing, ANN, and RNN-LSTM. *Nonlinear Processes in Geophysics* 27(3): 373–389. doi:10/ghnj74.

Doan, N. A. K.; Polifke, W.; and Magri, L. 2019. A Physics-Aware Machine to Predict Extreme Events in Turbulence. *arXiv:1912.10994* .

Ehrendorfer, M. 2011. *Spectral numerical weather prediction models*. Society for Industrial and Applied Mathematics. ISBN 978-1611971989.

Frank, M.; Drikakis, D.; and Charissis, V. 2020. Machine-Learning Methods for Computational Science and Engineering. *Computation* 8: 15.

Jaeger, H. 2001. The "Echo State" Approach to Analysing and Training Recurrent Neural Networks - with an Erratum Note. *German National Research Center for Information Technology GMD Technical Report* 148(34): 13.

Lorenz, E. N. 1963. Deterministic nonperiodic flow. *Journal of the atmospheric sciences* 20(2): 130–141.

Martinuzzi, F. 2020. SciML/ReservoirComputing.Jl. SciML Open Source Scientific Machine Learning.

Raissi, M.; Perdikaris, P.; and Karniadakis, G. E. 2019. Physics-Informed Neural Networks: A Deep Learning Framework for Solving Forward and Inverse Problems Involving Nonlinear Partial Differential Equations. *Journal of Computational Physics* 378: 686–707. ISSN 0021-9991. doi:10/gfzbvx.

Saha, S.; Moorthi, S.; Pan, H.-L.; Wu, X.; Wang, J.; Nadiga, S.; Tripp, P.; Kistler, R.; Woollen, J.; Behringer, D.; et al. 2010. The NCEP climate forecast system reanalysis. *Bulletin of the American Meteorological Society* 91(8): 1015–1058.

Saha, S.; Moorthi, S.; Wu, X.; Wang, J.; Nadiga, S.; Tripp, P.; Behringer, D.; Hou, Y.-T.; Chuang, H.-y.; Iredell, M.; et al. 2014. The NCEP climate forecast system version 2. *Journal of climate* 27(6): 2185–2208.

Wikner, A.; Pathak, J.; Hunt, B.; Girvan, M.; Arcomano, T.; Szunyogh, I.; Pomerance, A.; and Ott, E. 2020. Combining Machine Learning with Knowledge-Based Modeling for Scalable Forecasting and Subgrid-Scale Closure of Large, Complex, Spatiotemporal Systems. *Chaos: An Interdisciplinary Journal of Nonlinear Science* 30(5): 053111. ISSN 1054-1500. doi:10/ggxrjq.