Minimal Reservoir Computing

Haochun Ma[1,2], Davide Prosperino[1,2], Christoph Räth[3]

contact christoph.raeth@dlr.de

[1]Ludwig-Maximilians-Universität München, Faculty of Physics
[2]Allianz Global Investors, risklab
[3]Deutsches Zentrum für Luft- und Raumfahrt (DLR), Institute for AI Safety and Security

presented at **ITISE** on 15th-17th July 2024 in Gran Canaria, Spain

Abstract

Minimal reservoir computing is a novel machine learning technique for predicting complex systems. It simplifies the classical reservoir computing approach by eliminating the need for randomness. Instead of a random embedding, it embeds its input into a high-dimensional space structurally. Additionally, the reservoir is no longer a random graph, but a block-diagonal matrix. The reservoir states are simply evolved linearly, and the nonlinearity is pushed to the readout layer. The output is then a linear combination of the reservoir states, determined by a simple linear regression. In this work we simplify the initial approach even further by utilizing a diagonal matrix as a reservoir, effectively dropping the notion of it. In classical reservoir computing the reservoir states bear no obvious interpretation, as they are a representation of the reservoir at the time. However, in this setup the reservoir states are interpretable and represent nonlinear combinations of input space. Using a small number of data points, it is possible to fully capture the dynamics of the attractor in the short- and long-term using this simple setup. In this work we show that the weights of the linear regression can be utilized to derive recursive equations of complex systems. We analyze the stability of the discovered equations with regards to various hyperparameters. In the end we test its applicability on financial markets and study the modelling of interest rates using this model.

Minimal reservoir computing

Minimal reservoir computers [1] are a new flavor of reservoir computers with the goal of removing any randomness out of the process.

Embedding of input data

$\mathbf{W}_{\text{in}}$ is constructed in such a way, that each combination of the input dimension $\underline{u}$ is fed into the reservoir separately. The allowed coordinate combinations of the input are encoded in the hyperparameter η_{in}. $\underline{w}$ is a weights vector of length b assigning each coordinate a specific weight.

$$\mathbf{W}_{\text{in}} = \begin{pmatrix} \underline{w} & \underline{0} & \underline{0} \\ \underline{0} & \underline{w} & \underline{0} \\ \underline{0} & \underline{0} & \underline{w} \\ \underline{w} & \underline{w} & \underline{0} \\ \underline{w} & \underline{0} & \underline{w} \\ \underline{0} & \underline{w} & \underline{w} \\ \underline{w} & \underline{w} & \underline{w} \end{pmatrix}$$

Reservoir and its reservoir states

The reservoir $\mathbf{A}$ is a block-diagonal matrix of ones with size b scaled to a target spectral radius of ϱ^*. This way, each block $\mathbf{J}_i$ can be directly mapped to a feature.

$$\mathbf{A} = \frac{\varrho^*}{b}\begin{pmatrix} \mathbf{J}_1 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{J}_2 & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{J}_{n_f} \end{pmatrix} = \frac{\varrho^*}{b}\begin{pmatrix} \mathbf{J}_x & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{J}_y & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{J}_{x+y+z} \end{pmatrix}$$

The reservoir states $\underline{r}$ are evolved **linearly**, without any nonlinear activation function.

$$\underline{r}(t+1) = \mathbf{A}\,\underline{r}(t) + \mathbf{W}_{\text{in}}\,\underline{u}(t)$$

Creating the prediction

We use generalized reservoir states $\tilde{\underline{r}}$, where we append the states exponentiated to orders up to a nonlinearity degree η.

$$\tilde{\underline{r}} = \left(\underline{r}^1, \underline{r}^2, \cdots, \underline{r}^{\eta-1}, \underline{r}^{\eta}\right)^{\mathsf{T}}$$

Each dimension of the predicted state $\underline{y}$ is then a linear combination of the generalized reservoir state $\tilde{\underline{r}}$.

$$\underline{y}(t+1) = \mathbf{W}_{\text{out}}\,\tilde{\underline{r}}(t+1)$$

Training the readout

The readout matrix $\mathbf{W}_{\text{out}}$ is trained the usual way by stacking the reservoir states into a matrix and using ridge regression with a regularization parameter β.

$$\mathbf{W}_{\text{out}} = \mathbf{U}\,\tilde{\mathbf{R}}^{\mathsf{T}}\left(\tilde{\mathbf{R}}\,\tilde{\mathbf{R}}^{\mathsf{T}} + \beta\mathbf{I}\right)^{-1}$$

Results

Predictions are accurate up to a few Lyapunov times on short time scales and accurately model the climate for long time scales.

For some combinations of hyperparameters the short-term prediction of the Lorenz system is correct for up to **ten** Lyapunov times using only 500 training points [2].

Special parametrization $b = 1,\ \varrho^* = 0$

For a block size of $b = 1$ and a target spectral radius of $\varrho^* = 0$ the architecture simplifies drastically. The input matrix $\mathbf{W}_{\text{in}}$ becomes a binary matrix, and the reservoir $\mathbf{A}$ disappears.

The reservoir now has no memory and no abstract embedding. The reservoir state now simply represents the nonlinear combinations of the input dimensions. Using $\underline{u} = (x,\ y,\ z)$ we arrive at the following states.

$$\underline{r} = \begin{pmatrix} x \\ y \\ z \\ x+y \\ \vdots \\ x+y+z \end{pmatrix} \qquad \tilde{\underline{r}} = \begin{pmatrix} x \\ \vdots \\ x+y+z \\ x^2 \\ \vdots \\ (x+y+z)^2 \\ \vdots \\ (x+y+z)^{\eta} \end{pmatrix}$$

We note that this parametrization makes this method similar to a parametrization of next generation reservoir computing [3]. However, a detailed comparison is still required.

Training

We perform a simple ridge regression on the generalized reservoir states.

Results

We are able to predict the Lorenz system for up to five Lyapunov times by simply regressing the exponentiated sums of the coordinates against the next predicting state (Fig. 1).

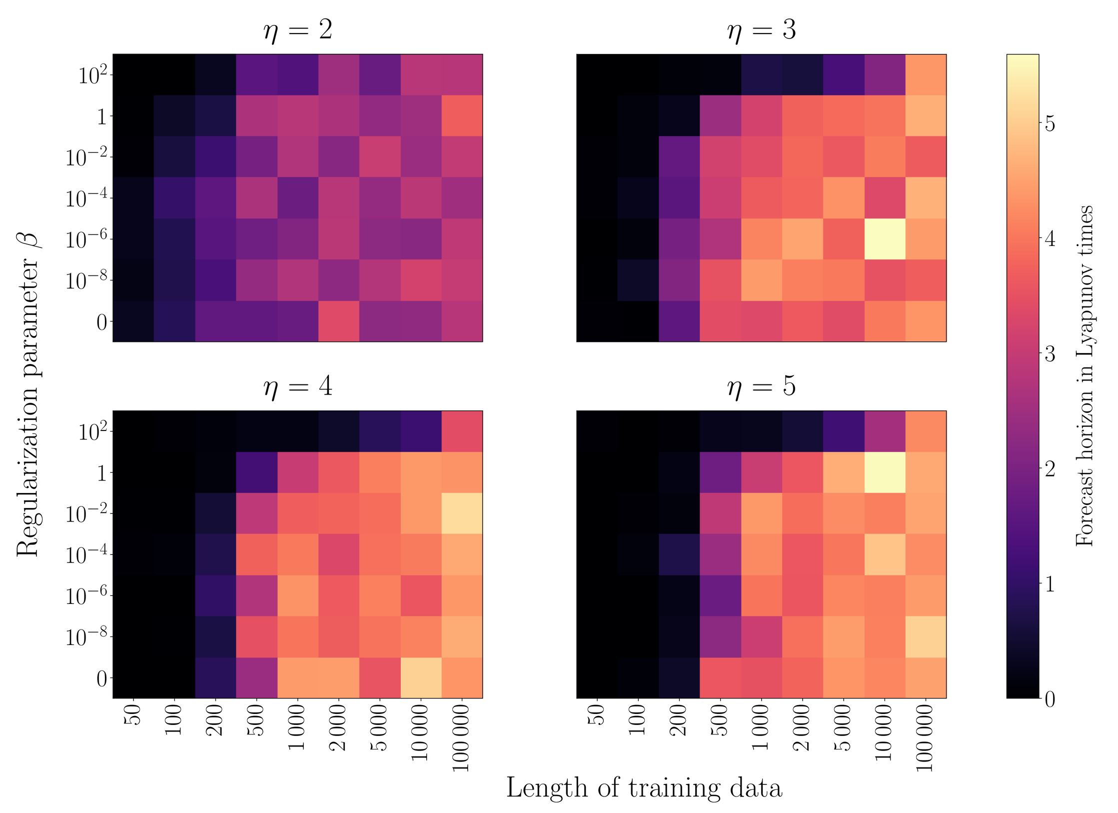


Fig. 1: Short-term prediction performance for the Lorenz system (σ=10, ϱ=28, β=8/3) using Δt=0.01 for different choices of hyperparameters. We performed each run ten times and reported the average.

Regarding the long-term climate of the predictions, we find that if the short-term prediction is reasonable, the attractor is usually reproduced.

Additionally, we do not need a "warm-up"/synchronization phase as we, effectively, do not have a reservoir anymore.

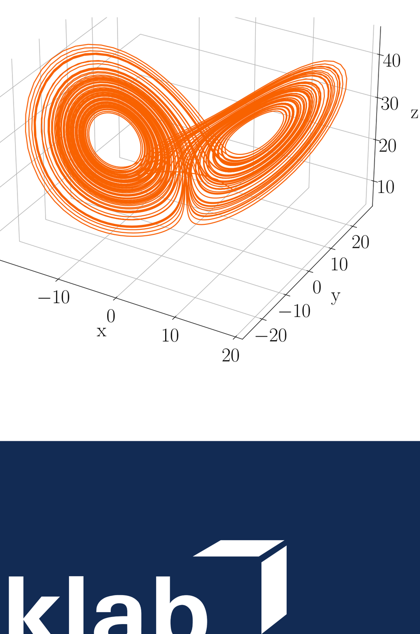
Reconstructing the recursive equations

$\mathbf{W}_{\text{out}}$ shows how each exponentiated input coordinate is going to be weighted for each output dimension. After the terms are expanded, we arrive at an iterative set of equations describing the system at hand. Each step corresponds to a time of Δt.

$$\underline{x}_{n+1} = f(\underline{x}_n)$$

Fig. 2 shows the application of the iterative equations for a library up to **fifth order**, and we can see the Lorenz attractor being reproduced with the Lyapunov exponent and correlation dimension being appropriately hit.

Fig. 2: Long-term prediction of the Lorenz system trained on 500 training points with hyperparameters $(\eta_{\text{in}},\ \eta,\ \beta) = (3,\ 5,\ 0.01)$. The iterated equations have a Lyapunov exponent of $\lambda = 0.80$ and a correlation dimension of $C = 1.96$.



Stability of parameters

Fig. 3 shows the stability of parameters for different runs, hence different positions on the attractor.

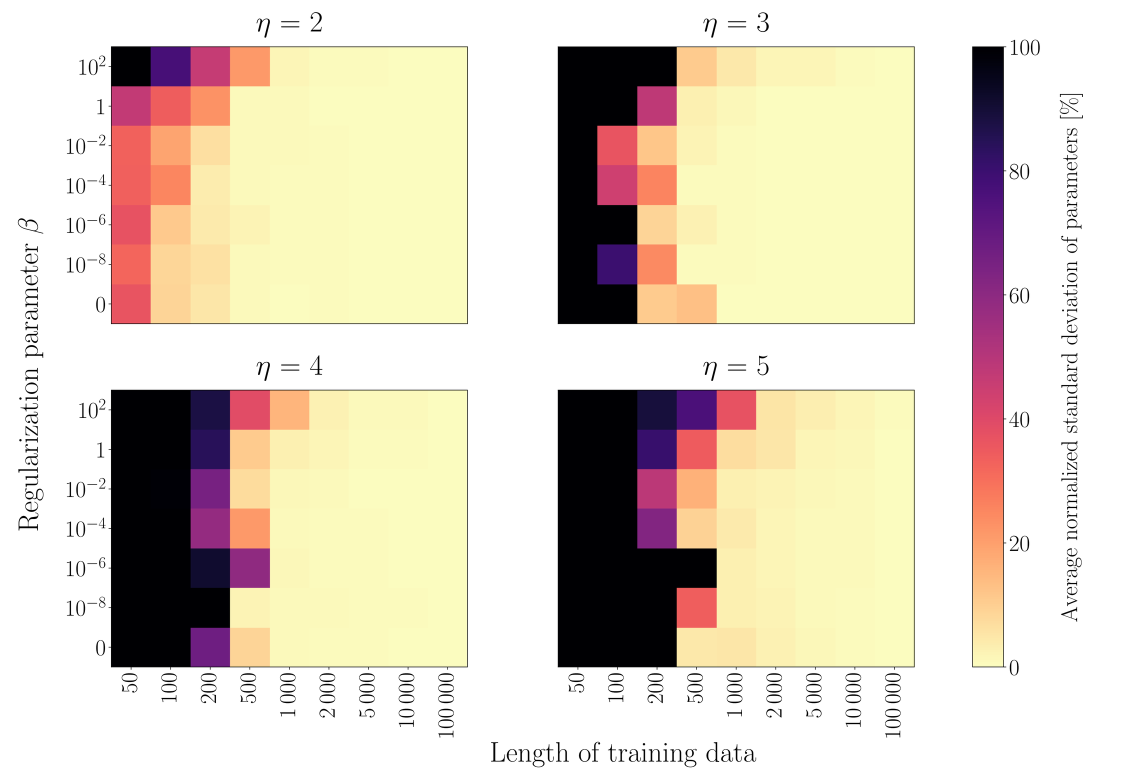


Fig. 3: This figure shows the **weighted** average of the normalized standard deviation across all parameters for a specific set of hyperparameters. The average is weighted by the magnitude of the parameter. The setup of Fig. 2 is used.

In Fig. 3, we reported the weighted average of the normalized standard deviation, because the simple mean is large and unstable across different hyperparameters.

This means that a lot of parameters with small numerical values have a large standard deviation. However, the few parameters with a large numerical value have a small relative standard deviation and are therefore stable across runs.

Since Fig. 2 shows that the prediction works across a wide range of hyperparameters, we may conclude that the exact numerical value of some parameters does not matter for the dynamic of the attractor and multiple parametrizations can build the Lorenz attractor [4].

Yield curve modelling

We model the daily U.S. Treasury yield curve by fitting our model to 250 days of data, the equivalent of one trading year. This optimization is performed over a rolling window with a step size of 20 trading days.

We then analyze the weights/terms of the resulting fit. The sum of the absolute values of the linear terms is used as an indicator of linear strength, while the sum of the absolute weights of the nonlinear terms is used as an indicator of nonlinear strength. Both values are normalized to have a mean of one.

For Fig. 4, we performed a grid search and selected the five models with the best performance.

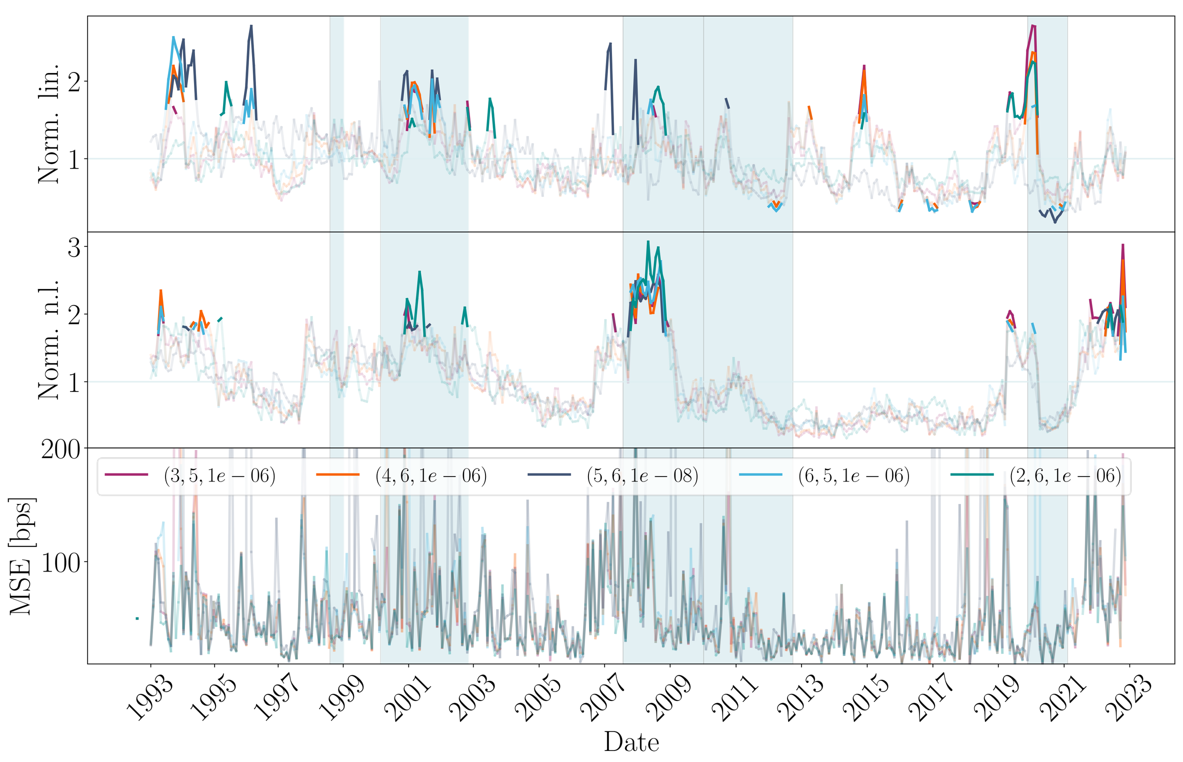


Fig. 4: **Top figure:** normalized sum of absolute values of linear terms. **Middle figure:** normalized sum of absolute values of nonlinear terms. **Bottom figure:** average mean squared error on the training set. The legend encodes the hyperparameters of the model: $(\eta_{\text{in}},\ \eta,\ \beta)$. Light blue shades indicate times of crises. Vibrant colors indicate a deviation from the mean of one and a half standard deviations.

During times of crisis, we observe that both the linear and nonlinear relative strengths deviate from normal levels. This observation is consistent with findings from [5-6], which noted that such a relationship was only significant for the nonlinear attributes of stock return time series.

References

[1] Ma *et al.*, Scientific Reports **13**, 12970 (2023).
[2] Ma, Dissertation, Ludwig-Maximilians-Universität München (2024).
[3] Gauthier *et al.*, Nature Communications **12**, 5564 (2021).
[4] Prosperino *et al.*, currently in review at Journal of Physics: Complexity.
[5] Haluszczynski *et al.*, Physical Review E **96**, 062315 (2017).
[6] Ma *et al.*, currently in review at Chaos, preprint on arXiv: 2312.16185.