SDSU | College of Sciences

# Initial Findings of Distributed Reservoir Computation

David A. Kaauwai

## Introduction

Time series computation is an important part of decision making processes in autonomous and self-aware computational systems. The ability of a computational system to accurately infer and predict possible future internal and external states directly informs that system's ability to perform the tasks that it may be designed to do. Reservoir computers are exceptionally lightweight, space efficient, computationally efficient, and quick to train in contrast to transformers and other neural networking models. Another property that has not been explored as of yet is the reservoir model's generalizable accuracy across similar tasks with the ability to separate the static reservoir portion of computation from the trained readout layer and how these properties may be leveraged in edge computation.

Research into computational applications between edge and terminal devices on the network has grown commensurate with the growth of ubiquitous computation and automation. A generalizable model that may be instantiated in resource constrained contexts and can be trained with very few data points is a valuable resource to have at the edge of the network for autonomous system decision-making and general time-series prediction. This initial work begins to evaluate how reservoir computers may fit into computational applications at the edge of the network between edge and terminal devices. Figure 1 shows a diagram of the architecture of the proposed and evaluated system.

## Methodology

This project investigates a distributed reservoir computing framework in which the data generation process and training of the readout layers and the reservoir state updates are executed on separate ROS 2 nodes each housed in separate docker containers on a simulated network. As illustrated, the architecture of the system is constructed from two primary components:

**Agent Node** — Generates time-series data by numerically integrating a chosen nonlinear dynamical system. It publishes batches of state values to the network and receives processed reservoir outputs.
**Edge Node** — Implements the reservoir computer. It receives state data from the agent, performs reservoir state evolution using a fixed recurrent weight matrix, and transmits the updated reservoir state back to the agent.

Two continuous chaotic systems of differential equations, the Röestler and Lorenz-96 systems, integrated with the fourth-order accurate Runge-Kutta 45 integrator simulate a live stream of data. Each system is solved with parameters and initial conditions that are known to generate chaotic attractors.

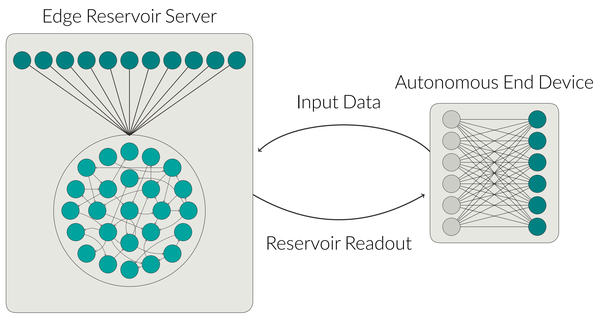


Figure 1. A graphical representation of a distributed reservoir computer.
An autonomous end device sends raw sensor data to the edge reservoir server to be processed and receives the readout from the reservoir. The end device completes the computation of that data through the trained weights of its output layer.

## Reservoir Configuration

The reservoir implementation used in this project follows the classical Echo State Network formulation. An initial hyperparameter search established that that the mean square error of prediction would not improve with an increased number of parameters in the reservoir while increasing the number of parameter correlated with an increase in round trip time of each message between the reservoir and end device container. Given the results of our hyper-parameter search, the reservoir dimension is set to 1024, the leak rate to 0.1, and a spectral radius to 1.1. Each reservoir is instantiated with the same seed (42) and random number generation algorithm to ensure that all evaluations are instantiated with the same parameter weights. The weight matrix is initialized using a power-law distribution to create a spectrum of weights with a heavy-tailed structure. We choose to square the value of the output of the activation function on at random as the reservoir is iterated forward. This initialization method and activation function modification increase the dynamical richness of the recursive algorithm and is consistent with reservoir designs used in recent literature. Only the readout layer is trained using a linear ridge optimization, consistent with standard reservoir computing practice [1, 2, 3].

## Results

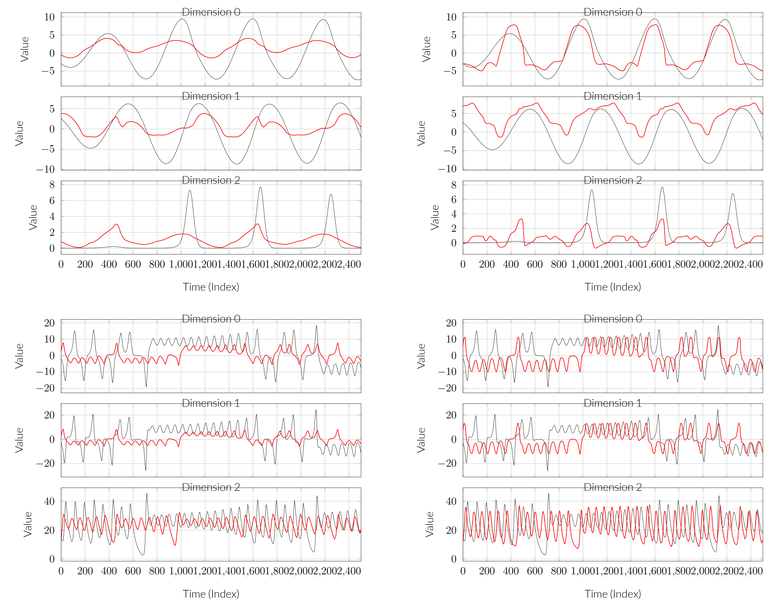


Figure 1 plots the results of 2500 time step predictions of the Lorenz and Rössler systems by a reservoir predictor with a readout layer trained for the respective system. Figure 1 plots the results of 2500 time step predictions of the Lorenz and Rössler systems by a reservoir predictor with a readout layer trained for the respective system. Note that both reservoir predictors have comparable accuracy with a low RMSE (see Table **??**). The reservoir preserves the characteristic chaotic behavior of both systems without exceeding the bounds of either system, and the predictors achieve accuracy on par with state of the art reservoir computer implementations [1, 2].

### A highlighted block containing some math

A different kind of highlighted block.

$$\int_{-\infty}^{\infty} e^{-x^2} dx = \sqrt{\pi}$$

Interdum et malesuada fames $\{1,4,9,\ldots\}$ ac ante ipsum primis in faucibus. Cras eleifend dolor eu nulla suscipit suscipit. Sed lobortis non felis id vulputate.

#### A heading inside a block

Praesent consectetur mi $x^2+y^2$ metus, nec vestibulum justo viverra nec. Proin eget nulla pretium, egestas magna aliquam, mollis neque. Vivamus dictum $\mathbf{u}^\mathsf{T}\mathbf{v}$ sagittis odio, vel porta erat congue sed. Maecenas ut dolor quis arcu auctor porttitor.

#### Another heading inside a block

Sed augue erat, scelerisque a purus ultricies, placerat porttitor neque. Donec $P(y \mid x)$ fermentum consectetur $\nabla_x P(y \mid x)$ sapien sagittis egestas. Duis eget leo euismod nunc viverra imperdiet nec id justo.

## Nullam vel erat at velit convallis laoreet

Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Phasellus libero enim, gravida sed erat sit amet, scelerisque congue diam. Fusce dapibus dui ut augue pulvinar iaculis.

| First column | Second column | Third column | Fourth |
|---|---|---|---|
| Foo | 13.37 | 384,394 | $\alpha$ |
| Bar | 2.17 | 1,392 | $\beta$ |
| Baz | 3.14 | 83,742 | $\delta$ |
| Qux | 7.59 | 974 | $\gamma$ |

Table 1. A table caption.

Donec quis posuere ligula. Nunc feugiat elit a mi malesuada consequat. Sed imperdiet augue ac nibh aliquet tristique. Aenean eu tortor vulputate, eleifend lorem in, dictum urna. Proin auctor ante in augue tincidunt tempor. Proin pellentesque vulputate odio, ac gravida nulla posuere efficitur. Aenean at velit vel dolor blandit molestie. Mauris laoreet commodo quam, non luctus nibh ullamcorper in. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos.

Nulla varius finibus volutpat. Mauris molestie lorem tincidunt, iaculis libero at, gravida ante. Phasellus at felis eu neque suscipit suscipit. Integer ullamcorper, dui nec pretium ornare, urna dolor consequat libero, in feugiat elit lorem euismod lacus. Pellentesque sit amet dolor mollis, auctor urna non, tempus sem.

## References

[1] Andrea Ceni and Claudio Gallicchio. Edge of stability echo state networks. August 2023. doi: 10.48550/ARXIV.2308.02902.

[2] Daniel J. Gauthier, Erik Bollt, Aaron Griffith, and Wendson A. S. Barbosa. Next generation reservoir computing. *Nature Communications*, 12(1), September 2021. ISSN 2041-1723. doi: 10.1038/s41467-021-25801-2.

[3] Grigoryeva Lyudmila and Juan-Pablo Ortega. Echo state networks are universal. June 2018. doi: 10.48550/ARXIV.1806.00797.



dkaauwai0821@sdsu.edu