The Oscillatory Reservoir Computer

Walter Peregrim

University of Maryland

MSML 699: Independent Study in Machine Learning

Table of Contents

1 Abstract

To investigate the effect of mechanical forces on cell to cell communication we propose the Oscillatory Reservoir: a novel reservoir computing architecture which allows the client to expose a portion of the reservoir synapses to a configurable, periodic force. The oscillatory reservoir is trained on a dynamical system which switches between two different attractors, the limit torus and the Thomas, via a square wave bifurcation as a function of time. During inference the reservoir approximates future trajectories based on a relatively small portion of attractor data. The quality of these predictions, which can be improved via hyperparameter tuning, inform us on the reservoir's ability to distinguish between the two attractors. We benchmark against a classic reservoir computer with the same exact setup so as to illuminate the effect of oscillatory entrainment on network robustness. From these results we concluded that imposing a macro-oscillatory stimulus on a reservoir computer improves its long term memory which subsequently allows it to successfully differentiate between the two attractors during inference.

2 Introduction

2.1 Neural Oscillations

Brainwaves, or neural oscillations, refer to recurring or rhythmic patterns of neural activity occurring in the central nervous system. The presence of oscillatory activity in the brain is widespread across various levels of organization, and it is believed to hold crucial significance in the processing of neural information[1]. Its generation within neural tissue can result from various mechanisms, including interactions between neurons or processes taking place within individual neurons. These oscillations may manifest as fluctuations in the membrane potential of neurons or as regular sequences of action potentials, subsequently leading to the rhythmic stimulation of postsynaptic neurons. Understanding the generation and functions of neural oscillations as well as their role in memory and learning constitutes a significant focus of research within the field of neuroscience[2]. To narrow the scope of this question we only investigate the role of neural oscillations in a single cognitive task, memory storage, which refers to the process of stabilizing and strengthening memories over time. Oscillatory patterns of neural activity can drive the induction of long-term potentiation (LTP) and long-term depression (LTD) at synapses, which are mechanisms underlying memory formation and consolidation[3].

2.2 Neural Synchronization

Synchronization in the brain, or neural synchronization, describes the coordination and alignment of spiking patterns across neuron populations, a process which contributes to the generation and modulation of neural oscillatory patterns. It is speculated that harmonized neural activity plays a crucial role in promoting effective communication and information processing throughout the brain[4]. Synchronization theory is an extensive research topic with too many methods and formulations to cover here, instead we focus on one approach: oscillatory entrainment. Oscillatory entrainment in the brain refers to the synchronization or alignment of neural oscillations with external rhythmic stimuli or internal oscillatory patterns. It is the phenomenon where the timing and frequency of neural oscillations become influenced by and aligned with the timing and frequency of an external stimulus or an ongoing internal oscillation[5]. When the brain is exposed to a rhythmic sensory stimulus, such as a visual or auditory stimulus with a specific frequency, the neural activity in corresponding brain regions can synchronize and align with the frequency of the stimulus. This alignment is known as phase-locking or phase synchronization, where the phase of the neural oscillations becomes entrained or locked to the phase of the external stimulus[6]. Although it is important to note that neural entrainment occurs in the presence of internal stimuli as well, a popular example being brain-generated modulatory patterns such as theta, alpha, and gamma brainwaves. These oscillations entrain and coordinate the firing of neurons involved in memory encoding and consolidation, thereby promoting the integration and storage of information[7].

2.3 Synaptic Modulation

Synaptic modulation plays a crucial role in neural synchronization by influencing the strength and timing of synaptic connections between neurons. It refers to the dynamic adjustment of synaptic transmission efficacy, which can impact the synchronization of neural activity. By modulating synaptic strength, neurons can enhance or weaken the influence they have on each other. This modulation can be achieved through various mechanisms, such as changes in neurotransmitter release, alterations in receptor sensitivity, or modifications in the quantity and properties of synaptic receptors. These synaptic modifications can occur on different timescales, ranging from rapid changes within milliseconds to longer lasting alterations. Stronger synaptic connections facilitate the transmission of signals and promote synchronization, allowing for coordinated firing among groups of neurons. Conversely, weakened or inhibited synaptic connections can disrupt synchronization and reduce the coherence of neural activity. The precise timing of synaptic events is also critical for synchronization, as it determines the phase relationships between neurons and their ability to generate coherent oscillatory patterns. By adjusting the timing of synaptic inputs, synaptic modulation can promote or inhibit synchronization depending on the specific circumstances[8].

2.4 Mechanobiology

Mechanobiology is an emerging field of science at the interface of biology, engineering, chemistry and physics that investigates the role of mechanical forces and mechanical properties in cognitive processes. This type of force exerted on a biological neural network can influence its ion channel activity, cell to cell transmission, and synaptic formation/plasticity[10]. Therefore it stands to reason that to achieve rhythmic synchronization across a population of neurons, one can enforce a periodic mechanical stimulus to synaptically modulate the network. While the modulatory capability of mechanical signals on neuronal membranes and membrane channels has been researched extensively, the mechanical aspects of neural network operation are not well studied.

2.5 Chaotic Dynamical Systems

We are learning the characteristics of nonlinear dynamical systems. A dynamical system is a mathematical object that models the evolution of some physical phenomena over time by describing its instantaneous state as a point in an ambient space. All dynamical systems can be classified as either stochastic or deterministic according to their evolution rule which, based on the current state, describes its future state. The deterministic system is a system whose future state does not depend on randomness, intuitively this means the model always produces the same output for a given state. The systems studied in chaos theory, chaotic systems, are deterministic, which means their future state can theoretically be predicted. But in practice, despite having deterministic dynamics, a chaotic system is only predictable for a certain period of time, after which it starts exhibiting complex, unpredictable behavior. These systems are characterized by their sensitivity to initial conditions, which means that small perturbations in the initial state of the system can lead to significantly different outcomes. Dynamical systems in nature generally arise from dissipative systems; the driving force and dissipation counterbalance each other, which kills off initial transients and settles the system into its typical behavior. The subset of the state space that the system's trajectory approaches, known as the attractor, corresponds to the time interval over which the system exhibits normal behavior. If this behavior is chaotic, i.e. if the underlying dynamical system is chaotic, the attractor is strange.

Consider performing chaotic time series inference on the Mackey-Glass attractor, introduced as a standard benchmark system for temporal prediction studies[12]. The MGS, as illustrated in the following figure, is known for its rich dynamics, including the emergence of chaotic behavior for certain parameter values.

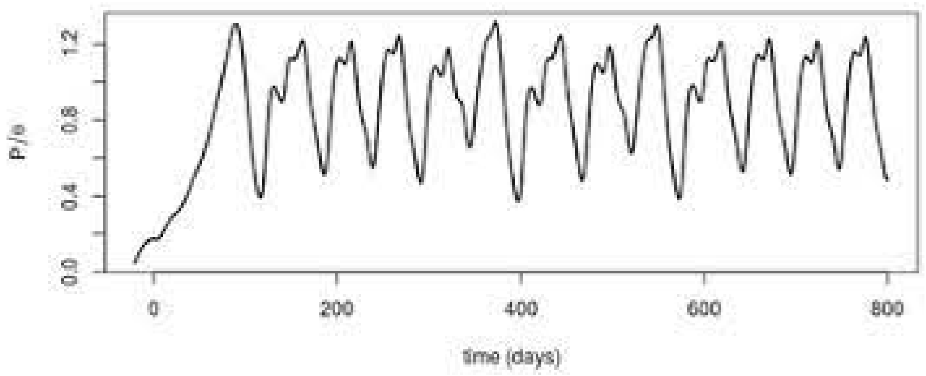


Figure 1: Mackey Glass System

However, we want to predict trajectories in the three dimensional space, not 1D, which means we need a different attractor than Mackey-Glass. This leads us to several 3D chaotic systems as introduced in [13] and [14], but we narrow our focus to only the Lorenz and Thomas attractors.

2.6 Reservoir Computing

To model an interconnected network of neurons we employ reservoir computing (RC), a computationally-efficient machine learning framework derived from recurrent neural network theory which is tailored for learning the dynamics of large, chaotic systems. Reservoir computers are a more efficient and scalable way to process spatiotemporal data than the standard RNN because its key distinction is that the input and hidden layers are fixed and only the output layer is trained. The hidden layer, aka the reservoir, represents a biological neural network which typically consists of a large number of nodes with random connections between them. The following diagram points out architectural parallels between the two computing machines, which we use as a basis for continuing with this black-box computational scheme.

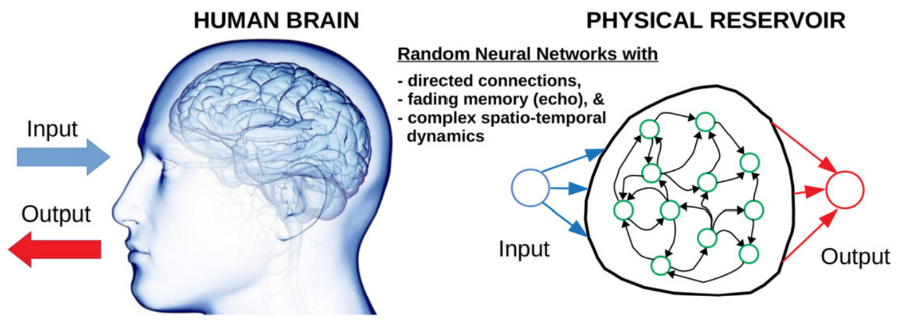

HUMAN BRAIN
PHYSICAL RESERVOIR
Random Neural Networks with
- directed connections,
- fading memory (echo), &
- complex spatio-temporal dynamics
Input
Output
Input
Output

Figure 2. The human brain, as well as a physical reservoir can be seen as a random neural network with directed connections, fading memory, complex spatio-temporal dynamics. Memory and spatio-temporal dynamics are properties of the underlying substrate/material (e.g., in the case of the brain it is an electrolytic environment in which complex, spatially non-uniform electrochemical reactions occur. In a physical reservoir, different from the brain, training is done in the output layer, which is not a part of the reservoir.

There are several types of reservoir computers, two of practical importance to us are the echo state network (ESN) and the liquid state machine (LSM). These denote the artificial and spiking formulations of a sparse (typically 1% connectivity) reservoir computer, respectively. Before the invention of the ESN, large reservoir computers were rarely used in practice because their training algorithms were slow and error-prone, which was due to the complexity of adjusting numerous connections[16]. This is why a sparse reservoir computer is ideal for modeling the characteristics of complex dynamical systems. Even though ESNs are tailored for learning behavior of strange attractors, their prediction accuracy is generally low without hyperparameter optimization. Building an RC involves initializing ~10 tunable hyperparameters whose values, configured by the client *a priori*, significantly impact network performance. The first few hyperparameter combinations used when tuning are rarely optimal, despite this there is a methodical approach to RC design, better expounded in[11].

3 Data & Data Analysis

3.1 Background

3.1.1 Lorenz System

The Lorenz system, first examined by mathematician and meteorologist Edward Lorenz, consists of a set of ordinary differential equations. It is renowned for exhibiting chaotic solutions under specific parameter values and initial conditions. The Lorenz attractor, a collection of chaotic solutions within the Lorenz system, holds significant implications in real-world scenarios, giving rise to the concept known as the "butterfly effect" in popular media. This effect highlights the unpredictability of chaotic systems, as various initial chaotic conditions evolve in phase space without ever repeating, emphasizing the inherent unpredictability despite their deterministic nature. Due to the continuous amplification of chaos within such systems, long-term predictions become highly challenging. When represented in phase space, the shape of the Lorenz attractor itself can be observed to bear resemblance to a butterfly[13].

However the attractors are very trivial, as seen in Figure 3 – either the butterfly wing, or a stable limit cycle around its perimeter, or decay into either centers of the wings. Trajectories that decay to a single point mean that any dynamics converge to a flat line over time, which is uninteresting, and the other two (strange attractor vs limit cycle about its perimeter) are too similar.

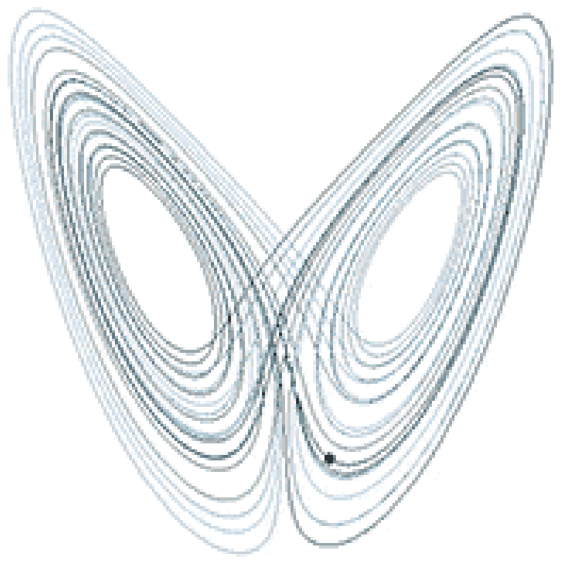


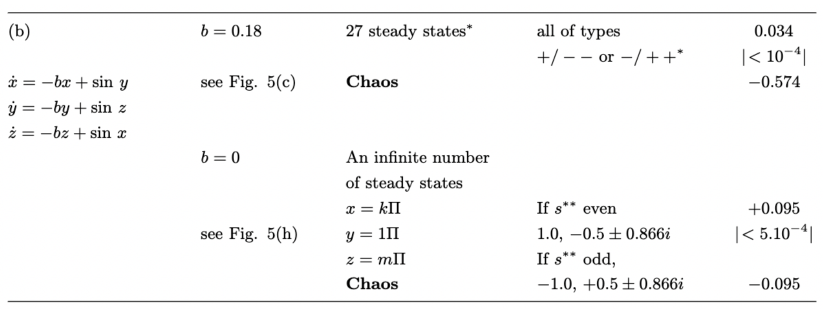
Figure 3: Lorenz Attractor

3.1.2 Thomas Attractor

The Thomas attractor, discovered by Ray Thomas, is a turbulent dynamic system characterized by intricate and unpredictable dynamics. It operates in three dimensions, and can be considered a general class of cyclically symmetric attractors---that is, they're invariant under coordinate cyclic permutations. When visualized, the Thomas attractor unveils an intricate and elaborate pattern in three-dimensional space. The system's trajectory traces a convoluted and non-repetitive path, forming a fractal-like structure. The shape of the attractor is highly sensitive to the initial conditions and parameter values, meaning slight alterations in these values can lead to significantly distinct trajectories. This sensitivity contributes to the unpredictable nature of chaotic systems.Beyond its intriguing mathematical properties, the Thomas attractor finds practical applications in various fields such as physics, engineering, and computer science. It serves as a model to comprehend and simulate complex phenomena characterized by chaotic behavior. The attractor's intricate structure and dynamics offer valuable insights into the nature of chaos and the challenges associated with predicting long-term outcomes in nonlinear systems. This behavior is governed by a set of differential equations that describe the evolution of these variables over time[14]. The equations describing the Thomas are:

$$
\begin{aligned}
\frac{dx}{dt} &= \sin(y) - bx \\
\frac{dy}{dt} &= \sin(z) - by \\
\frac{dz}{dt} &= \sin(x) - bz
\end{aligned}
$$

In these equations, x, y, and z represent the system's coordinates, while t denotes time. The parameter 'a' influences the system's behavior and can be adjusted to generate different types of attractors. The last term represents damping, so the "conservative" limit, b=0, represents conservation of energy. Despite its simplicity, it's a crazy system. Below b=0.3, which corresponds to the threshold of a Hopf bifurcation (that is, a nice periodic orbit), it undergoes numerous crises (whereby existing attractors keep colliding with different attractors to generate new dynamics, as the bifurcation parameter b is decreased). In some cases, up to 6 different chaotic systems can coexist to form the attractor (see attached papers).



(b)	$b = 0.18$	27 steady states*	all of types	0.034
			$+/--$ or $-/++$*	$\lvert < 10^{-4}\rvert$
$\dot{x} = -bx + \sin y$	see Fig. 5(c)	**Chaos**		-0.574
$\dot{y} = -by + \sin z$				
$\dot{z} = -bz + \sin x$				
	$b = 0$	An infinite number		
		of steady states		
		$x = k\Pi$	If s^{**} even	$+0.095$
	see Fig. 5(h)	$y = 1\Pi$	$1.0,\ -0.5 \pm 0.866i$	$\lvert < 5.10^{-4}\rvert$
		$z = m\Pi$	If s^{**} odd,	
		Chaos	$-1.0,\ +0.5 \pm 0.866i$	-0.095

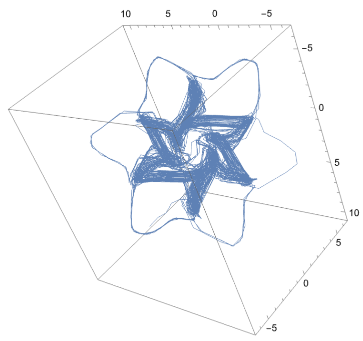


Figure 4: Thomas Attractor

As b approaches 0, it undergoes yet another bifurcation wherein every point is an unstable steady attractor, and thus the trajectory follows deterministic fractal Brownian motion, and therefore anomalous diffusion. Normally, Brownian motion is probabilistic, but nothing about our system is---our system is completely deterministic, i.e. the next point in phase space of x,y, and z, are uniquely defined by the differential equations above. Below is a gif as the bifurcation parameter evolves from b=0.001 to b=0.399 (so pretty much b=0 and b=0.4). You can see the initial ergodic fractal motion, which spontaneously bifurcates to a chaotic attractor that changes in orientation and size, and then finally, the system degrades to a small circle (Hopf bifurcation) until it decays to a single point.

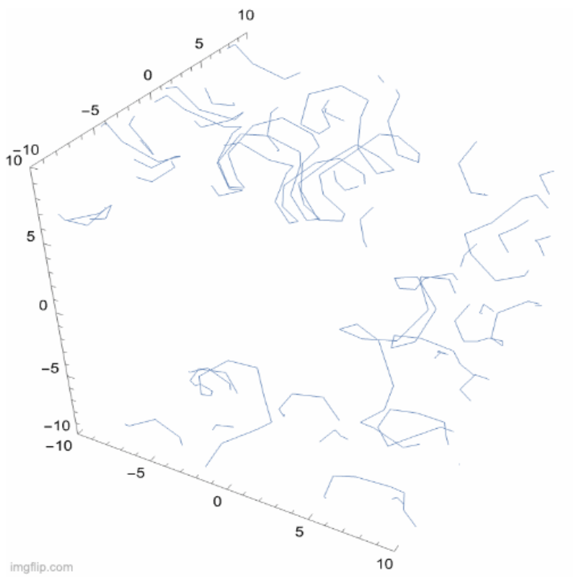


Figure 5: Thomas w/ an evolving bifurcation parameter

So the question of robustness can be rephrased as follows: Normally, robustness is quantified as the stability of predictions in the presence of noise. In the case of image classification, this just means classifying a noisy image. In the case for time series, this normally means adding noise to the state variables i.e. x, y, and z. But they still keep b constant. In our case, we can fluctuate b around a range of values, and ask---can the reservoir still predict the general attractor (after all, a reservoir trained solely on one value of b will outperform our case, but we're not concerned with that---we're concerned with how well it's able to interpolate the general dynamics of a system). To fluctuate the bifurcation parameter a bit more, or a bit faster, we changed b so that rather than being a cosine, it is a square wave. This is so that we can simply try storing 2 different attractors only (instead of everything at once as a cosine), as piecewise attractors, and simply see if the reservoir can spit out the proper attractor. We have chosen a simple limit torus (a blurred ring) and the strange attractor as the 2 attractors to test.

3.2 Data Description

3.2.1 1 Dimensional Data Description

Each attractor is simulated for 500 frames, before switching to the next attractor (i.e. b is held constant for some value for 500 frames, then changed suddenly to a different value for 500 frames, then back again to the first value for 500, etc.). This training data can be found in *cyclic_square.csv*. Because this is much easier for the reservoir to distinguish theoretically, we can reduce the training data to 15k frames. Note that the 1D plot of the training dataset above only shows the first 3000 timesteps, this is just to illustrate the attractor switching. The limit torus occurs at b = 0.29; hence, why the first validation datafile is named *cyclic_val_b_0_29.csv*. The strange attractor happens at b=0.15, and has the corresponding file name of *cyclic_val_b_0_15.csv*.

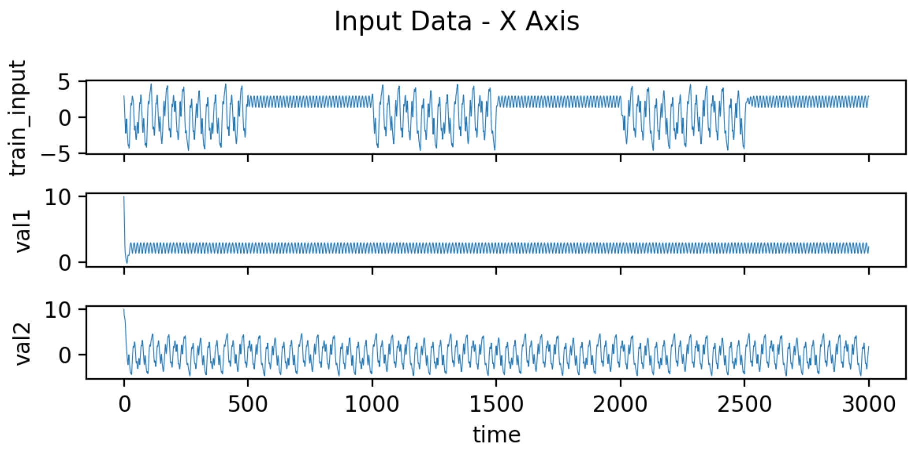


Figure 6: cyclic_square.csv

During the burning periods, also referred to as transient lengths, the reservoir synchronizes to a different dataset (either a new experiment or different parameter regime) by discarding the states in X. During the cooling periods the reservoir is either training or inferencing, depending on the dataset being evaluated. Unlike the training dataset, the validation datasets can each only have one burn period which starts at t=0 because the reservoir predicts the attractor dynamics for all the post-burn timesteps.

3.2.3 2 Dimensional Data Description

The same switching between attractors from the one dimensional representation of the data can be observed in the two dimensional plots below.

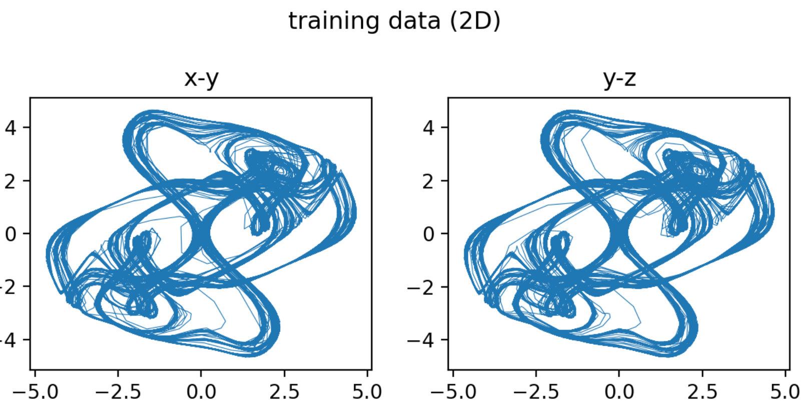


Figure 5: Training Data

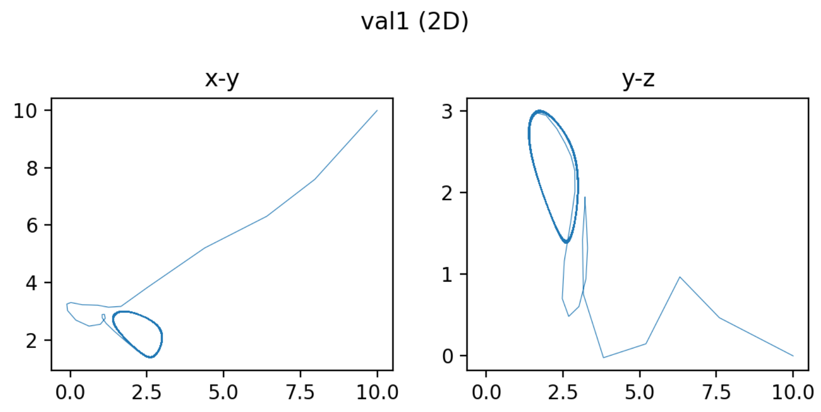


Figure 6: First Validation Dataset (b=0.29)

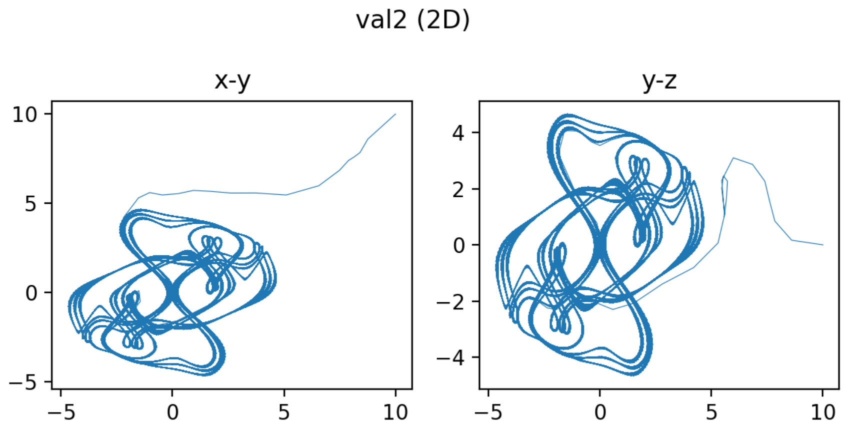


Figure 7: Second Validation Dataset (b=0.16)

4 Oscillatory Reservoir Walkthrough

4.1 Model Architecture

One important mechanism that allows neurons to synchronize their firing patterns and create coherent oscillations is entrainment, specifically entrainment by a periodic force, a phenomenon derived from synchronization theory. We start with the vanilla RC architecture which maps a lower dimensional input signal to a higher dimensional one. Since the oscillatory reservoir takes the spatial attractor data as input, we generate dynamical system trajectories as output.

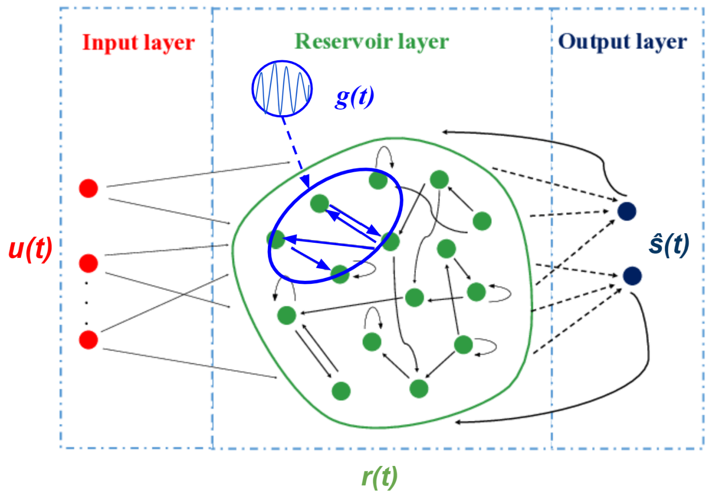


Figure 10: Oscillatory Reservoir Computer

The corresponding reservoir dynamics are defined in the equation below,

$$r(t + \Delta t) = (1 - \alpha)r(t) + \alpha tanh(Ar(t) - Am * g(t) + Win * u(t) + b1) \qquad (1)$$

$$g(t) = (MS/2)(1 + sin(\square t + \phi)) \qquad (2)$$

The output is found in the same way as the standard RC, by applying a linear transformation to the reservoir state, as seen in the operation below:

$$\hat{s}(t) = Wout * r(t) + c \qquad (3)$$

Omega is the angular frequency therefore its unit is radians per second and its period is (2π/ω). Omega is fixed across the whole modulation population, phase is random.

4.2 Loss Function

4.2.1 Discrimination Loss Equation

The qualitative metric we use to determine the robustness of our model is how well it distinguishes between the two chaotic attractors, similar to a binary classification task. Both the prediction vector and ground truth vector are traversed point for point and the distance to the nearest point in the other vector is computed. In other words, we first iterate over all spatial elements in the ground truth vector and for each one the distance to the nearest point in the predictions array is computed, and then this step is repeated except with swapped vectors (iterate over predictions and find nearest ground truth point). But we don't care about every single time step, only care about how the trajectories converge. Thus we only consider the last third of the ground truth data and predictions when computing the loss and drawing the visualizations. Next, the distances are summed, squared, and then averaged over the number of validation steps, and the resulting scalar is returned as the loss. The following equation and subsequent animation further clarify how we measure the success of our model:

$$\mathcal{L}^{GT} = \frac{\sqrt{(\Delta DP)^2 + (\Delta PD)^2}}{n} \text{ ; } \mathcal{L}^{GT} = \text{distance to ground truth} \qquad (1)$$

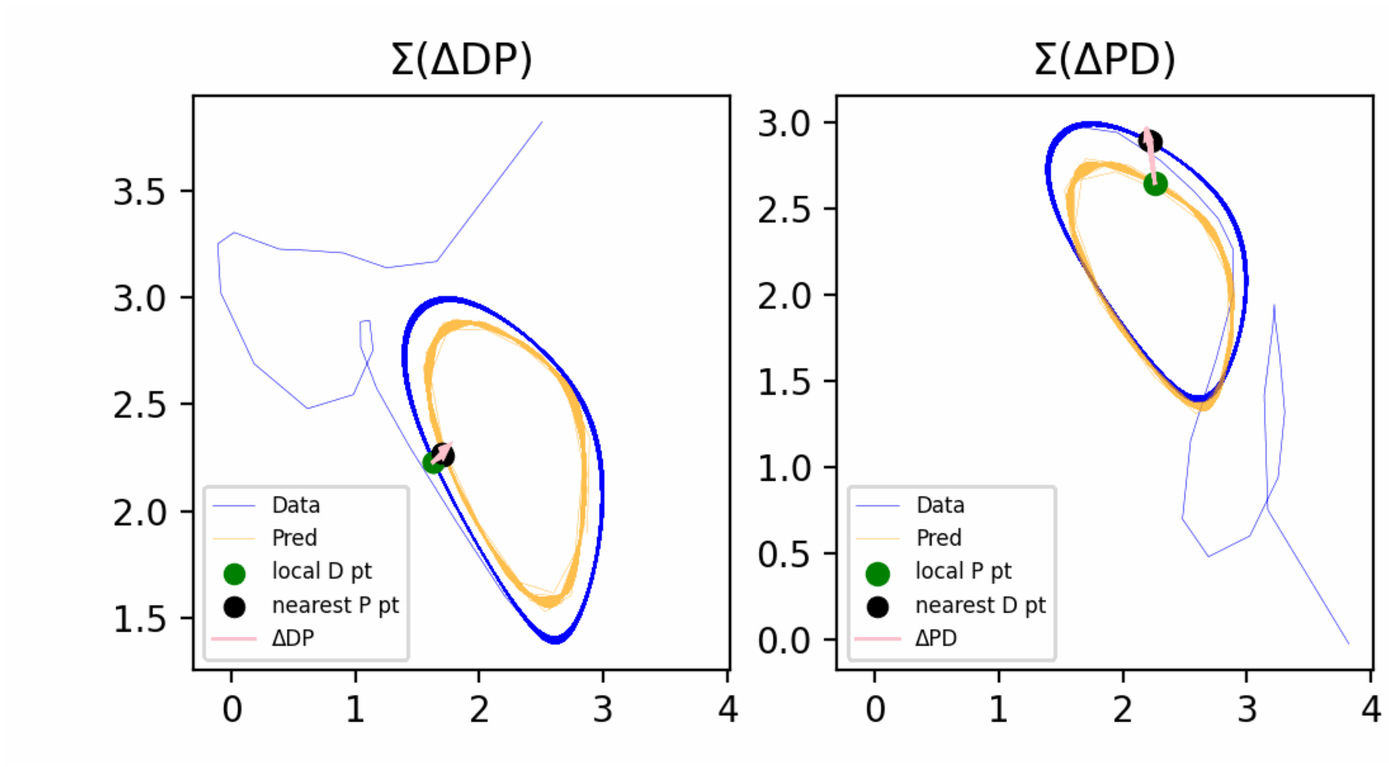


Figure 11: Animation of Loss Function

However this metric is not good enough, so we added another term to the loss function so that it better aligns with its qualitative goal of measuring our model's ability to distinguish between attractors. The new term is the reciprocal of the loss function in (1) to the incorrect chaotic attractor, like so:

$$\mathcal{L} = \mathcal{L}^{GT} + \frac{1}{\mathcal{L}^{AT}} \text{ ; } \mathcal{L}^{AT} = \text{distance to “alternative truth”} \qquad (2)$$

The updated loss function is twice as informative as the previous because it contains information about the similarity of the predictions to both attractors. We cannot truly infer whether or not our model distinguishes between two classes unless we understand the relationship of the predictions to both labels. The two term equation can be looked at as insurance that if the disparity of the predictions with respect to one class does not tell us enough, then the disparity to the other class will. To determine whether or not the model distinguishes between attractors, we compare the computed loss to a discrimination threshold specified by the client, which defaults to 2. If the loss is above the threshold, it does not distinguish between the attractors, and vice versa.

4.2.2 The Reflection Corner Case

Even though this metric does a good job of quantifying the discrimination capabilities of our reservoir, it actually doesn't cover a particular case where some large perturbation sways the predicted trajectory toward the other basin of attraction. This second limit cycle exists because of the point reflection symmetry: (x, y, z) -> (-x, -y, -z), which is visualized in part a) of Figure 13. To ameliorate this issue we added a thresholding to the cost function such that if either $\mathcal{L}^{AT}$ or $\mathcal{L}^{GT}$ is greater than 4, then the predictions are multiplied by -1 and the loss is recomputed. This reflects the predictions about y = -x, as observed in part b) of Figure 12 below. There are a few different scenarios with a loss greater than 4 in which the 2d limit torus does not reside in the bottom left quadrant for both the x-y and y-z plots. But we do not consider these predictions as correct because the equations of the chaotic attractor do not permit this kind of symmetry. So an additional clause was added to the loss function which reflects the predictions back to their original coordinate space if, and only if, the new loss is less than 2.

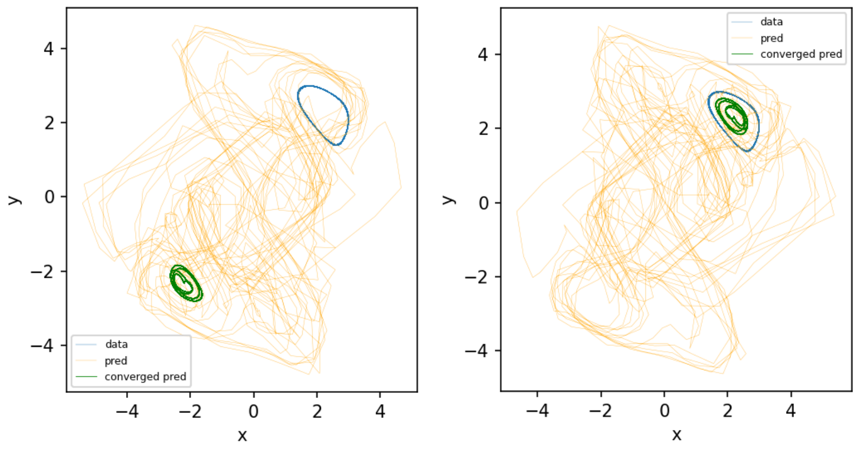


Figure 12: The Corner Case; a) problem, b) solution

4.2.3 The Single Point Corner Case

The trajectory predictions for both the limit cycle and the chaotic attractor will sometimes converge to a single point instead of following a path. When predicting the Thomas attractor this isn't much of an issue because the loss will remain high regardless of where the converged point lies. But for the limit cycle, a converged point within the area of torus, as seen in Figure 15, will result in a low loss for that prediction. Assuming the chaotic attractor prediction is accurate, the current loss function returns a value just above the discrimination threshold, so technically this case is already solved. However, the loss should be a lot higher because this convergence to a single point is also another attractor for the Thomas system for a different bifurcation value.

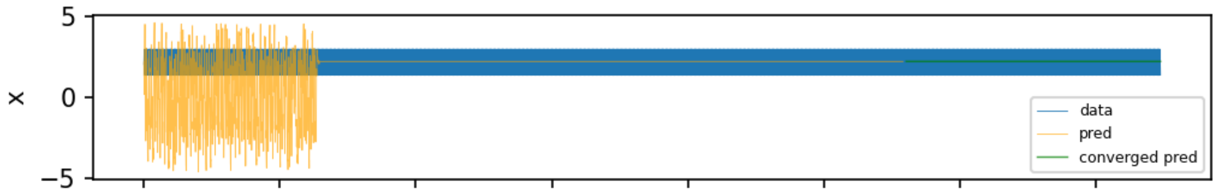


Figure 15: Limit Torus 1D Predictions

4.2.3 The Single Switch Corner Case

Since the relative timing of the trajectories is not taken into account when finding the distance to the nearest point (ΔDP and ΔPD calculations), only a small portion of the data will be looked at if a bifurcation switching occurs. Consider only the last quarter (green) of the limit torus predictions in Figure 13 below and take note of the bifurcation switch seen a few hundred timesteps in. This results in a low alternative truth loss (i.e. distance to thomas attractor), and consequently a high final loss (3-4), because the nearest spatial points are found in the timesteps before the flip. It is important to note that a bad low ground truth loss (< 2), specifically when predicting the Thomas attractor, can be a result of this corner case as it fails to catch the switch to limit torus. The current loss function cannot handle this corner case.

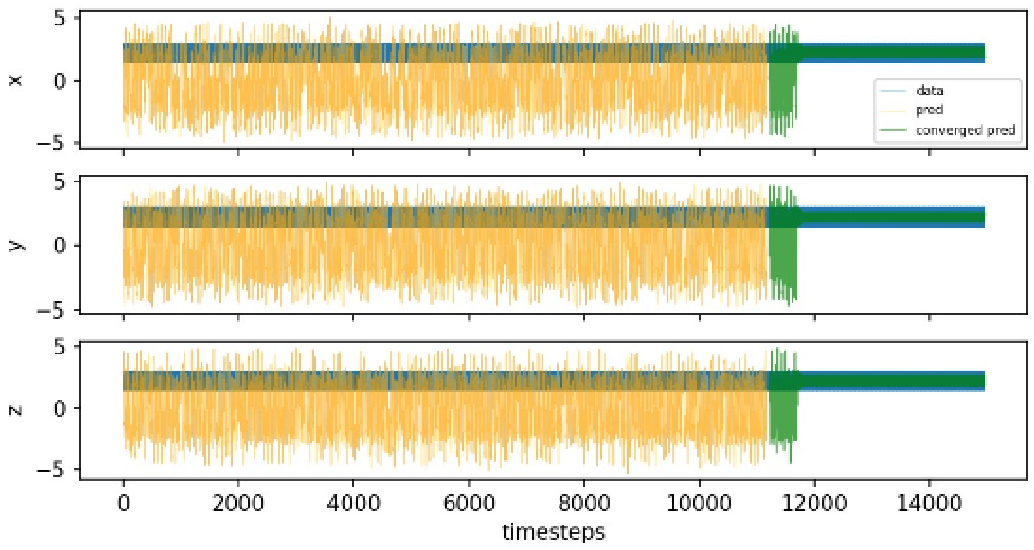


Figure 13: Limit Torus 1D Predictions

4.2.4 The Frequently Switching Corner Case

A bifurcation switch can occur more than just once, and if it happens frequently, as in Figure 14, then it's on the brink of discrimination. Even though the overall loss is low (<2) we regard it as non-discriminatory because the predictions don't "commit" to either a limit cycle or chaotic attractor. The current loss function cannot handle this corner case.

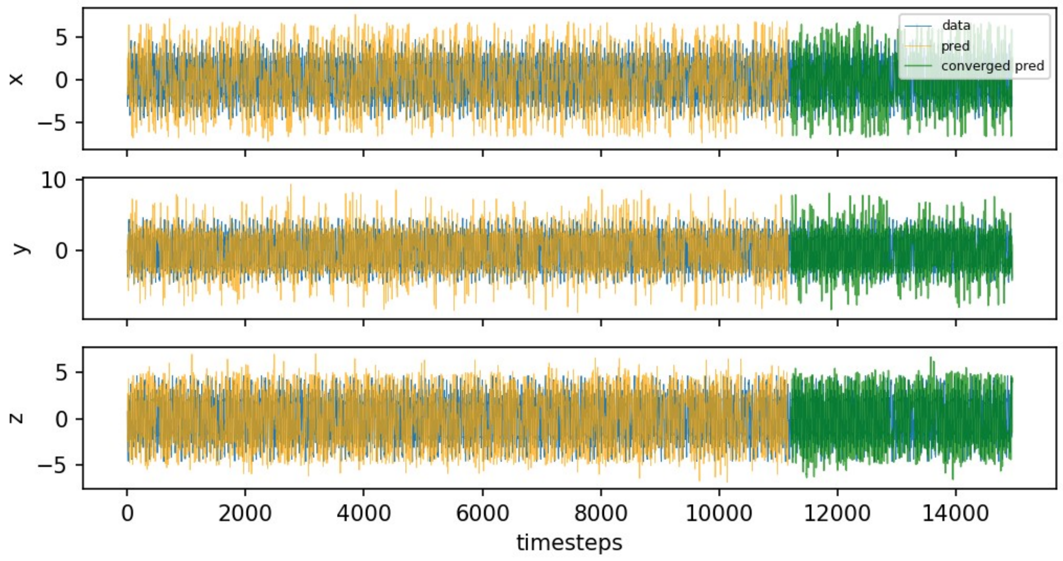


Figure 14: Thomas Attractor 1D Predictions

5 Tuning

5.1 Vanilla Reservoir Hyperparameters

We only focus on the four most important tunable reservoir hyperparameters out of roughly a dozen, each of which is expounded in the list below. The “optimal” combination of hyperparameters were obtained by testing different values and manually interpreting the ensuing model predictions. We employed hyperparameter optimization to verify our speculative results and came up with similar values/ranges. In the code the initial hyperparameter values default to our near optimal findings.

5.1.1 Reservoir Size

The crucial hyperparameter in determining the reservoir's extent is its size, often dictated by the neuron count. A bigger reservoir has the capability to encompass intricate dynamics, yet it could also introduce higher computational complexity and the risk of overfitting. Fine-tuning the reservoir size enables the discovery of an optimal equilibrium between model capacity and the ability to generalize. The default, and near optimal, reservoir size is 300.

5.1.2 Spectral Radius

The spectral radius is a hyperparameter that controls the amplification or attenuation of signals within the reservoir. It affects the reservoir's dynamic range and influences the memory capacity of the network. Tuning the spectral radius is crucial to achieve the desired memory capabilities and avoid signal saturation or vanishing effects. The near optimal spectral radius is 2.2. The default list of spectral radii to try is [1.0, 1.5, 2.0].

5.1.3 Connectivity

The connectivity pattern within the reservoir determines the information flow and the complexity of interactions between neurons. Hyperparameters related to reservoir connectivity, such as the sparsity or the type of connections (random, sparse, or structured), impact the reservoir's ability to process and store information. Tuning these hyperparameters can help optimize the reservoir's dynamics for a given task. The near optimal connectivity is 10.

5.1.4 Input Scaling

RC networks often require input scaling to ensure the input signals are in an appropriate range for the reservoir dynamics. The scaling factor affects how the input signals are mapped into the reservoir, its tuning can help avoid input signal saturation or loss of information. The near optimal input scaling is 1.8. The default list of input weights to try is [1.0, 1.5, 2.0, 2.5].

5.2 Oscillatory Reservoir Hyperparameters

We test various combinations of the 3 oscillatory hyperparameters, frequency, modulation population, and modulation strength on a number of data seeds, all of which can be configured by the client.

5.2.1 Frequency

The frequency is a hyperparameter that sets the "pace" of the rhythm imposed on the reservoir, in Figure 13 this controls the rate at which the blue links pulsate. A good rule of thumb is that omega's corresponding period should be ~10x as long as that of the inherent dynamics themselves, which gives us an optimal omega of $\frac{2\pi}{10*(50)}$, or 0.0126. The default list of frequencies to try is [0.005, 0.01, 0.02].

5.2.2 Modulation Strength

The amplitude of the periodic stimulus is set to half of this value, in Figure 13 this controls the thickness of the blue links. The near optimal modulation strength is 0.2. The default list of modulation strengths to try is [0.02, 0.03, 0.04].

5.2.3 Modulation Population

This parameter specifies the proportion of the synapses in the reservoir to modulate, in Figure 13 this controls the number of the blue links. The near optimal modulation population percentage is 0.1. The default list of modulation population percentages to try is [0.01, 0.02, 0.03].

5.2.4 Random Seeds

In our code the data seed is used to randomly create the sparse reservoir connection matrix, in Figure 13 this controls which links are blue. A unique data seed should result in unique reservoir behavior, so we test out a lot of seeds to ensure that our results aren't specific to a single random state. The range of data seeds to try defaults to (1-70).

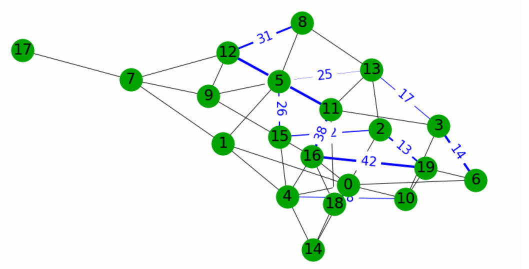


Figure 13: Example Oscillatory Reservoir Network

6 Results

6.1 Vanilla Reservoir Computer

To put the performance results of our model into context we benchmark it against two types of reservoir computers, the first of which is the contemporary reservoir computer who's synaptic weights are static during training and inference. For this learning model we tuned two hyperparameters, input weight and spectral radius. The results of this effort can be observed in Table 1 below. Out of the 12 hyperparameter scenarios, 10 of them saw no discrimination for any data seeds. The singular good seeds for the other 2 scenarios are covered by the frequently switching corner case and therefore can be disregarded.

		Input weight			
		1.0	1.5	2.0	2.5
Spectral radius	1.0	0/70	0/70	0/70	0/70
	1.5	1/70	0/70	0/70	0/70
	2.0	0/70	0/70	0/70	1/70

Table 1: Static Static

6.2 Fully Oscillatory Reservoir Computer

In a fully oscillatory reservoir computer the synaptic weights oscillate during both training and inference, and the validation results of this model can be viewed in the figure and tables below. Interestingly, one can observe the periodic force that was applied to the reservoir during inference in the output figures. Figure 15 displays predictions across the three omega values and it is clear that the period of the pulses is correlated with the frequency.

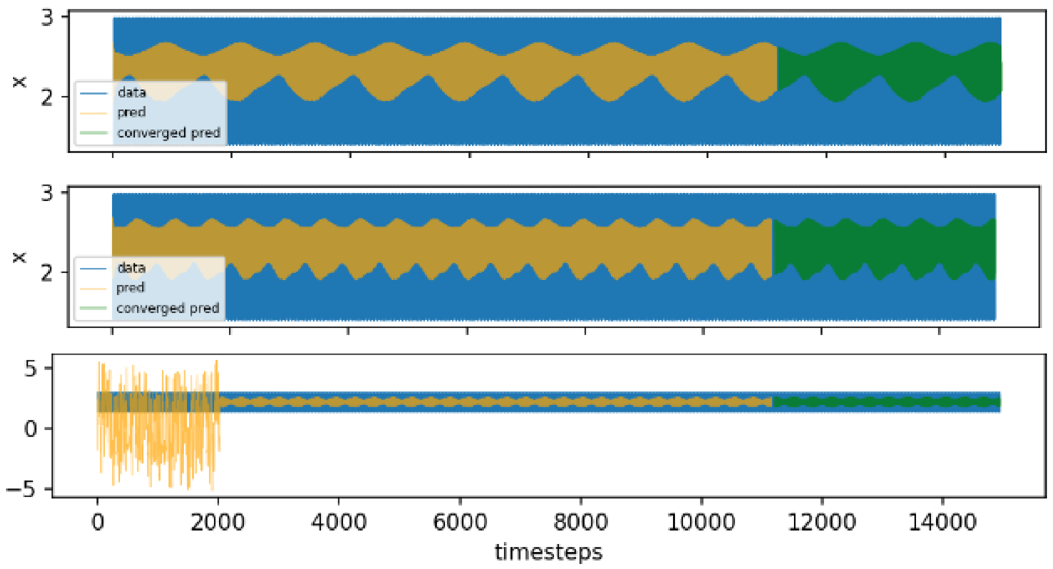


Figure 15: Omega=0.005 (top); 0.01 (middle); and 0.02 (bottom)

Moving along to the tables, when omega=0.005 the fully oscillatory reservoir performs roughly just as well as the static reservoir. For the next two omega values one can clearly see a jump in reservoir performance, but about half of the good results can be attributed to either the single switching or frequently switching corner cases. Therefore we consider this architecture to be slightly more robust than its static counterpart, but only just.

omega = 0.005		***modstrength***		
		0.2	0.3	0.4
modpop	0.1	0/70	0/70	0/70
	0.2	1/70	0/70	0/70
	0.3	0/70	0/70	0/70

Table 2

omega = 0.01		***modstrength***		
		0.2	0.3	0.4
modpop	0.1	3/70	0/70	2/70
	0.2	3/70	1/70	1/70
	0.3	1/70	0/70	0/70

Table 3

omega = 0.02		***modstrength***		
		0.2	0.3	0.4
modpop	0.1	1/70	1/70	1/70
	0.2	0/70	2/70	0/70
	0.3	0/70	0/70	0/70

Table 4

6.3 Our Reservoir Computer

In our reservoir architecture the synaptic weights only oscillate during training, not when inferencing. The results of training this model can be seen in the following tables. The first thing worth noting is that our oscillatory reservoir with near optimal oscillatory hyperparameters is able to discriminate over many more seeds than the fully static and fully oscillatory reservoirs. Second, and more importantly, our model maintains a high level of performance across a range of frequency values. Even with non-optimal omega values, the quality of results is still relatively evenly distributed across multiple hyperparameter combinations.

omega = 0.005		***modstrength***		
		0.2	0.3	0.4
modpop	0.1	8/70	3/70	2/70
	0.2	3/70	5/70	2/70
	0.3	1/70	1/70	1/70

Table 5

omega = 0.01		***modstrength***		
		0.2	0.3	0.4
modpop	0.1	6/70	4/70	2/70
	0.2	4/70	2/70	3/70
	0.3	3/70	2/70	1/70

Table 6

omega = 0.02		***modstrength***		
		0.2	0.3	0.4
modpop	0.1	5/70	3/70	2/70
	0.2	4/70	2/70	3/70
	0.3	2/70	1/70	2/70

Table 7

7 Discussion

7.1 Analysis

To better summarize the table results we created a histogram of the losses for each type of model, as illustrated in Figure 16. Although all three histograms look very similar due to the range of the y-axis, our model does perform considerably better than both the fully static and fully oscillatory reservoir.

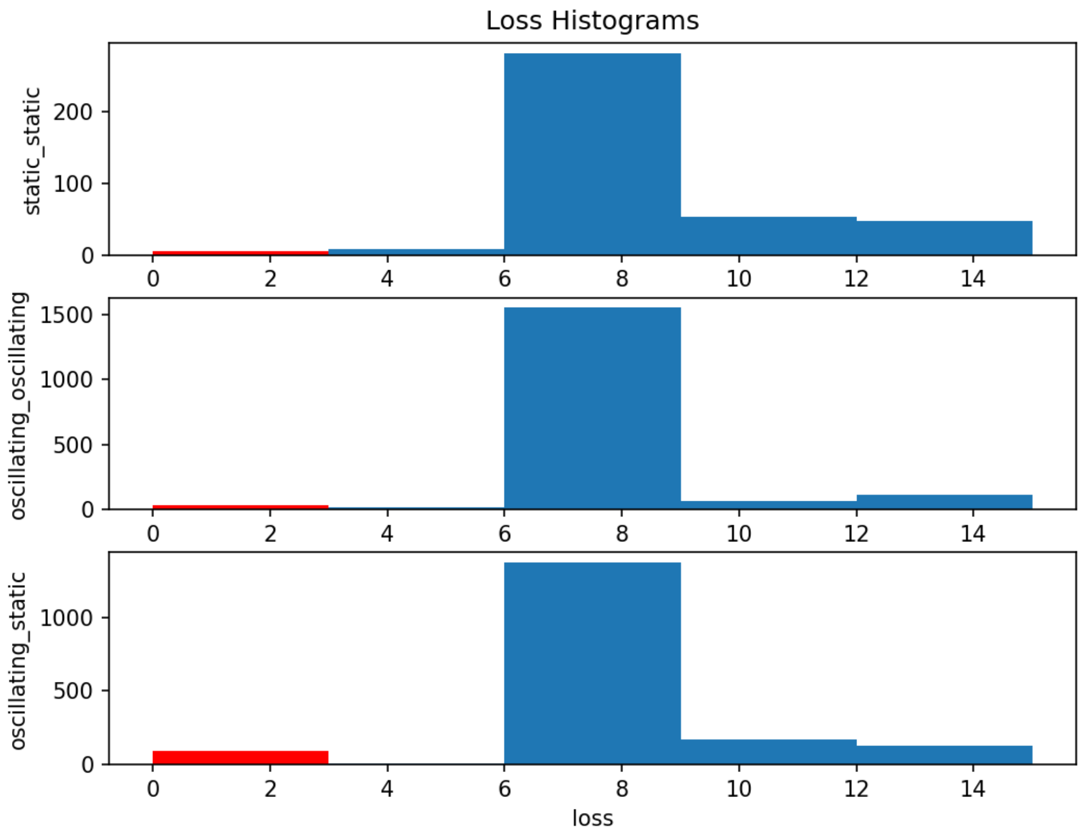


Figure 16: Loss Histogram

Therefore we can think of the periodic stimulus as a rhythm that is encoded into the reservoir which manipulates the manner in which it recalls information. Imposing a low frequency cadence on a portion of the model's synaptic links extends how long it remembers certain information, thus endowing the reservoir with long-term memory. We hypothesized that memories are encoded in the mechanochemical modulations applied to the reservoir computer, which our results support.

7.2 Future Work

redacted

6 Work Cited

1. Zamroziewicz, Marta K., and Aron K. Barbey. "Cognitive Neuroscience Meets the Community of Knowledge." *Frontiers*, https://www.frontiersin.org/articles/10.3389/fnsys.2021.675127/full. Accessed 21 June 2023.
2. Doelling, Keith B., and M. Florencia Assaneo. "Neural oscillations are a start toward understanding brain activity rather than the end." *PLOS*, 4 May 2021, https://journals.plos.org/plosbiology/article?id=10.1371/journal.pbio.3001234. Accessed 21 June 2023
3. Langille, Jesse J., and Richard E. Brown. "The Synaptic Theory of Memory: A Historical Survey and Reconciliation of Recent Opposition." *Frontiers*, 28 September 2018, https://www.frontiersin.org/articles/10.3389/fnsys.2018.00052/full. Accessed 21 June 2023.
4. Fusaroli, Riccardo. "Functional Synchronization: The Emergence of Coordinated Activity in Human Systems." *Frontiers*, 22 May 2017, https://www.frontiersin.org/articles/10.3389/fpsyg.2017.00945/full. Accessed 21 June 2023.
5. Pikovsky, Arkady, et al. *Synchronization: A universal concept in nonlinear sciences*, https://kyl.neocities.org/books/[TEC%20PIK]%20synchronization%20-%20a%20universal%20concept%20in%20nonlinear%20sciences.pdf. Accessed 21 June 2023.
6. Fell, Juergen, and Nikolai Axmacher. "The role of phase synchronization in memory processes." *nature reviews: neuroscience*.
7. Nyhus, Erika, and Tim Curran. "Functional Role of Gamma and Theta Oscillations in Episodic Memory." *NCBI*, 6 January 2010, https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2856712/. Accessed 21 June 2023.
8. Scheler, Gabriele. "Neuromodulation influences synchronization and intrinsic read-out." *NCBI*, https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6426090/. Accessed 21 June 2023.
9. Jutras, Michael J., and Elizabeth A. Buffalo. "Synchronous Neural Activity and Memory Formation - PMC." *NCBI*, 18 March 2010, https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2862842/. Accessed 25 July 2023.
10. Wang, J.H.C., Thampatty, B.P. An Introductory Review of Cell Mechanobiology. *Biomech Model Mechanobiol* 5, 1–16 (2006). https://doi.org/10.1007/s10237-005-0012-z
11. Lukosevicius, Mantas. "A practical guide to applying echo state networks." *Bernoulli Institute for Mathematics, Computer Science and Artificial Intelligence*, https://www.ai.rug.nl/minds/uploads/PracticalESN.pdf. Accessed 21 June 2023.
12. Jaeger, Herbert, and Harald Haas. "Harnessing Nonlinearity: Predicting Chaotic Systems and Saving Energy in Wireless Communication." *Columbia University*,

http://www.columbia.edu/cu/biology/courses/w4070/Reading_List_Yuste/haas_04.pdf. Accessed 21 June 2023.

13. Lu, Zhixin, and Jaideep Pathak. "Reservoir observers: Model-free inference of unmeasured variables in chaotic systems." *An Interdisciplinary Journal of Nonlinear Science*, https://pubs.aip.org/aip/cha/article/27/4/041102/322542/Reservoir-observers-Model-free-inference-of.
14. SPROTT, J. C., and KONSTANTINOS E. CHLOUVERAKIS. "LABYRINTH CHAOS." *International Journal of Bifurcation and Chaos*, https://sprott.physics.wisc.edu/pubs/paper302.pdf. Accessed 21 June 2023.