The Coupling Strength Is a Scale Parameter in Threshold Power-Law Reservoirs and Does Not Influence Training Accuracy

Wilten Nicola[1,2,3]
[1] Hotchkiss Brain Institute
[2] Department of Physics and Astronomy,
University of Calgary, Alberta, Canada
[3]Department of Cell Biology and Anatomy,
University of Calgary, Alberta, Canada

December 2, 2025

Abstract

In reservoir computing, the coupling strength of the initial untrained recurrent neural network (the reservoir) is an important hyperparameter that can be varied for accurate training. A common heuristic is to set this parameter near the "edge of chaos", where the untrained reservoir is near the transition to chaotic dynamics, and the chaos can be "tamed". Here, we investigate how the overall connectivity strength should be varied in threshold power-law recurrent neural networks, where the firing rate is 0 below some threshold of the current and is a power function of the current above this threshold. These networks have been previously shown to exhibit chaotic solutions for very small coupling strengths, which may imply that the chaos cannot be tamed at all. We show that for reservoirs constructed with threshold power-law transfer functions, if the reservoir can be trained for one single positive value of the initial reservoir coupling strength, then there exist networks with identical accuracy for all positive coupling strengths, implying that the chaotic dynamics can always be tamed or never be tamed. This is a direct consequence of the coupling strength of threshold power-law RNNs acting as a scale parameter that does not qualitatively influence the dynamics of the system, but only scales all system solutions in magnitude. This is independent of the power of the transfer function, with the exception of Rectified Linear Unit (ReLU) networks. This is in contrast with conventional RNNs/reservoirs employing sigmoidal firing rates, where the strength of the recurrent coupling in the initial reservoir determines the performance on different tasks during training and also influences the network dynamics explicitly.

Introduction

Recurrent Neural Networks (RNNs) serve as powerful tools in the pursuit of understanding brain operations. When these networks are untrained and coupled randomly, they provide valuable insights into how activity propagates through neural circuits [1, 2], how the presence of distinct subpopulations impacts overall network dynamics [3], how different time scales emerge in populations of neurons [4], how connectivity leads to synchronizability [5], and the types of transitions or bifurcations neural circuits may undergo as functions of a few key parameters [1, 6–8]. In addition to the analysis of emergent dynamics in untrained RNNs, a parallel approach of training RNNs to mimic the behaviours of real neural circuits has also recently emerged [9–12]. These RNNs are often trained with reservoir computing [13, 14], where the initially untrained RNN is set near the "edge of chaos", a parameter regime where the network dynamics are near a critical point at the onset of self-perpetuating firing rate fluctuations (chaos).

Recently, biologically plausible threshold power-law RNNs have been developed that consider non-sigmoidal firing rates. These networks have been studied with varying powers including

square-root firing rates which mimic the firing rate typically expected from Hodgkin-Class I neurons [15], supralinear firing rates which model responses of cells in the visual cortex [16–19], and Rectified Linear Unit (ReLU) firing rates which emerge for neurons with spike-frequency adaptation at their steady-state [20]. In parallel to the numerical simulations of these networks, variants of these networks have been analyzed from multiple perspectives including how they transition to hyperchaotic dynamics [6, 7, 21], how slow time scales emerge in these networks [22], how information gets encoded by these RNNs [23], along with analytical derivations of the moments of large-scale populations in [24, 25]. In the former cases, the majority of the networks were ReLU based, while supralinear firing rates were most notably analyzed in stabilized supralinear networks [16–19].

The work in [6, 7] demonstrates that for large threshold power-law RNNs, the chaotic dynamics of these networks emerge immediately for any non-zero coupling strength. Given the edge-of-chaos heuristic for conventional reservoirs, and the immediate onset of chaotic dynamics in threshold power-law RNNs, the question of whether or not threshold power-law RNNs are trainable with reservoir computing naturally emerges. Further, if threshold power-law RNNs are trainable, how should they be initialized in terms of the coupling strength hyperparameter?

Here, we show both analytically and numerically that threshold power-law RNNs operate differently from their classical sigmoidal cousins, but can be trained. First, we found that the coupling strength analytically has no impact on the potential training accuracy for any supervisor, for all non-RELU threshold power-law RNNs. This was a consequence of the fact that the coupling strength in these networks is actually a scale parameter and the network transfer functions are scale-invariant. The coupling strength does not qualitatively alter the dynamics of the reservoir at all, beyond scaling the amplitudes of the different solutions, along with their basins of attraction. We showed analytically and numerically that if a network could be trained for any non-ReLU threshold power-law firing rate for a single value of g with some level of test accuracy, then there exists a trained reservoir for any other value of g with identical accuracy. Further, we demonstrate numerically that threshold power-law RNNs can indeed learn target dynamical systems like oscillators and low-dimensional chaotic dynamics for powers less than 1.

Results

Consider the threshold power-law RNN given by the equations

$$\frac{dz_i}{dt} = -z_i + g\sum_{j=1}^{N}\omega_{ij}f(z_j), \quad i = 1, 2, \ldots N \tag{1}$$

$$f(z) = \begin{cases} z^k & z > 0 \\ 0 & z \leq 0 \end{cases}, \quad k > 0 \tag{2}$$

for any $k > 0$ (Figure 1A). The function $f(z_j)$ is typically interpreted as the firing rate for neuron j with some common examples being $k = \frac{1}{2}$ corresponding to the firing rate of the quadratic integrate-and-fire neuron [15], while $k = 1$ corresponds to the ReLU (Rectified Linear Unit) [26] and $k > 1$ corresponds to stabilized supralinear networks [16–19]. The variable z_j is sometimes interpreted as the current for neuron j. When $f(z)$ is a sigmoidal function and $g \propto \sqrt{N}^{-1}$, the network in (1) is of the class originally considered by [1]. For the classical rate networks with sigmoidal transfer functions considered in [1], namely, $f(z) = \tanh(z)$, chaotic solutions emerge as $N \to \infty$ for $g > \sqrt{N}^{-1}$, while the system's equilibrium point at $z = 0$ is asymptotically stable for $g < \sqrt{N}^{-1}$ and a sharp transition occurs for $g = \frac{1}{\sqrt{N}}$.

More recently, threshold power-law recurrent networks similar to equations (1)-(2) were considered in [7]. In [7], the authors conclude that for powers k where $k \leq \frac{1}{2}$, there exists no

stable-fixed point in a system similar to (1)-(2) as $N \to \infty$ and determine numerically that the system can be chaotic even for small values of g, which we reproduce here in Figure 1B-C. Both [7] and [6] also show that as $N \to \infty$, chaos emerges even for small coupling strengths ($g \ll \sqrt{N}^{-1}$), and thus chaos is likely the only solution for large N, and all $g \neq 0$.

We considered the non-sigmoidal firing rates in (1)-(2), which were more generally analyzed in [6, 7], for their potential use in reservoir computing. In many instantiations of reservoir computing, the weights of (1) are modified with a rank-m perturbation to stabilize the system onto the dynamics of some m-dimensional supervisor, $\hat{\boldsymbol{x}}(t)$

$$\frac{dz_i}{dt} = -z_i + g\sum_{j=1}^{N}\omega_{ij}f(z_j) + \sum_{l=1}^{m}\boldsymbol{\eta}_{lj}\boldsymbol{\phi}_{lj}f(z_j) \tag{3}$$

$$\hat{\boldsymbol{x}}(t) = \boldsymbol{\phi}^T f(\boldsymbol{z}) \tag{4}$$

where $\hat{\boldsymbol{x}}(t)$ is a m dimensional approximation to the target dynamics of $\boldsymbol{x}(t)$ and the encoders ($\boldsymbol{\eta}_i$) and decoders ($\boldsymbol{\phi}_i$) are $m \times 1$ vectors for each neuron (with index i). The initial weights, ω_{ij} are randomly generated from the Gaussian with mean 0, and standard deviation 1. The $\frac{1}{\sqrt{N}}$ that is conventionally present can be absorbed into the scalar g. Note that negative values of g typically need not be considered, as the sign of g can be absorbed into the weights ω_{ij}. In the case of weights drawn from random distributions with mean 0, as in [6, 7], the dynamics of the network are unaffected.

In matrix form, the reservoir is the initial nonlinear dynamical system given by the equation:

$$\frac{d\boldsymbol{z}}{dt} = -\boldsymbol{z} + g\boldsymbol{\omega}f(\boldsymbol{z}) \quad \text{(Initial Reservoir)} \tag{5}$$

where after successful training, the system's weights are perturbed with the low-rank matrix $\boldsymbol{\eta}\boldsymbol{\phi}^T$:

$$\frac{d\boldsymbol{z}}{dt} = -\boldsymbol{z} + \left(g\boldsymbol{\omega} + \boldsymbol{\eta}\boldsymbol{\phi}^T\right) f(\boldsymbol{z}), \quad \hat{\boldsymbol{x}} = \boldsymbol{\phi}^T f(\boldsymbol{z}) \quad \text{(Trained Reservoir)} \tag{6}$$

and $\hat{\boldsymbol{x}}(t) \approx \boldsymbol{x}(t)$. During learning, $\boldsymbol{\eta}$ is typically held fixed and is randomly generated at the start of a simulation, while $\boldsymbol{\phi}$ is learned in some way to minimize the following loss function (the mean-squared error):

$$L(\boldsymbol{\phi}, \boldsymbol{\eta}, g, \boldsymbol{\omega}) = \sum_{l=1}^{m}\int_0^T \left(\hat{x}_l(t') - x_l(t')\right)^2 dt' \tag{7}$$

although other losses may also be considered. Note that we've explicitly identified all parameters $(\boldsymbol{\phi}, \boldsymbol{\eta}, g, \boldsymbol{\omega})$ which determine the loss. We will interpret the loss in equation (7) as the test loss where the interval $[0, T]$ occurs after training has concluded (and $\boldsymbol{\phi}$ is held constant). There are many techniques to learn the decoder $\boldsymbol{\phi}$ either iteratively in an online setting (e.g. FORCE training [9, 10, 27] or Backprop through time [12]) or offline (e.g. Echo State Training [13, 28]), or more recently, through predictive alignment [11]. We do not specify how the decoder is learned in the analysis, but we use recursive least squares (RLS) here (Materials and Methods, [9, 27]).

When $f(z) = \tanh(z)$, typically g is selected near $g = \frac{1}{\sqrt{N}}$ as this is the "edge of chaos" for large networks, although the range of g required varies with the learning algorithm employed and the supervisor considered [9]. When the $\frac{1}{\sqrt{N}}$ term is absorbed into the weights, the edge of chaos occurs for $g = 1$ [1]. In this work, we seek to determine how g should be scaled for threshold power-law reservoirs, given previous reports of chaos for even small values of g for large networks [7]. Surprisingly, we find that g does not explicitly determine the loss, for any-sized network, as stated in the following theorem:

Theorem 1 *Suppose that the initial reservoir (5) is trained on some m-dimensional supervisor* $\boldsymbol{x}(t)$ *with some encoder/decoder pair* $\boldsymbol{\eta}$, $\boldsymbol{\phi}$ *for* $g = g^*$, *achieving a test-loss of* $L^* = L(\boldsymbol{\phi}, \boldsymbol{\eta}, g^*, \boldsymbol{\omega})$. *Further, assume that the threshold power-law considered is* $k \neq 1$ *(non-ReLU). Then for any* $\bar{g} > 0$, *there exists an encoder and decoder pair,* $\hat{\boldsymbol{\eta}}$, $\hat{\boldsymbol{\phi}}$ *with* $L(\hat{\boldsymbol{\phi}}, \hat{\boldsymbol{\eta}}, \bar{g}, \boldsymbol{\omega}) = L^*$

Before we prove Theorem 1, there are a few key points to note. First, if the initial reservoir can be trained with some $g = g^*$, then it can be trained for all g with equivalent accuracy $L = L^*$, so long as the correct encoder and decoder pair ($\hat{\boldsymbol{\eta}}$, $\hat{\boldsymbol{\phi}}$) can be found by the learning algorithm employed. Second, through the very straightforward derivation of Theorem 1, we will demonstrate that $\hat{\boldsymbol{\eta}}$, $\hat{\boldsymbol{\phi}}$ can be expressed in terms of $g, g^*, \boldsymbol{\eta}$, $\boldsymbol{\phi}$, and k. The third key point to note is that Theorem 1 implies that if a learning algorithm can find an optimum (local or global) low-rank perturbation matrix for a single value of the coupling strength g in (6), then this solution is also a local/global optimum for any g. Finally, the theorem does not specify how the initial reservoir weights $\boldsymbol{\omega}$ are generated, or even if the training was successful (L_0 is small). Thus, if the chaotic dynamics can be tamed for a single value of $g = g^*$, they can be tamed for all values of g. The reason this theorem can be so general is that it is a consequence of the following lemma:

Lemma 1 *Consider the threshold power-law recurrent neural network given by*

$$\frac{d\boldsymbol{z}}{dt} = -\boldsymbol{z} + g\boldsymbol{\omega} f(\boldsymbol{z}) \tag{8}$$

where $k \neq 1$. *Then (9) can be rescaled to the following system:*

$$\frac{d\boldsymbol{y}}{dt} = -\boldsymbol{y} + \boldsymbol{\omega} f(\boldsymbol{y}) \tag{9}$$

with $\boldsymbol{y} = g^{\frac{1}{k-1}}\boldsymbol{z}$ *where* k *is the power of the threshold power-law transfer function.*

The derivation of Lemma 1 is extremely straightforward:

$$\begin{aligned}
\frac{d\boldsymbol{y}}{dt} &= g^{(k-1)^{-1}}\frac{d\boldsymbol{z}}{dt} \\
&= g^{(k-1)^{-1}}(\boldsymbol{y}g^{-(k-1)^{-1}} + g\boldsymbol{\omega} f(\boldsymbol{y}g - (k-1)^{-1}) \\
&= -\boldsymbol{y} + g^{(k-1)^{-1}+1-k(k-1)^{-1}}\boldsymbol{\omega} f(\boldsymbol{y}) \\
&= -\boldsymbol{y} + \boldsymbol{\omega} f(\boldsymbol{y})
\end{aligned}$$

In plain terms, Lemma 1 states that the coupling strength, g, acts as a scale parameter for threshold power RNNs when $k \neq 1$. Thus, g does not qualitatively influence the network dynamics for all $g \neq 0$, and only scales the system's solutions. All solutions that exist for a particular $\boldsymbol{\omega}$ at some particular value of g exist for all values of g for that same ω, up to a rescaling factor. This has multiple implications, including the straightforward derivation of Theorem 1. First, if the system has chaotic solutions for a particular value of g^*, then it has (the same) chaotic solutions for all $g \neq g^*$ and these are rescaled versions of the chaotic solutions that exist for $g = g^*$. This was numerically confirmed in simulations (Figure 1D). There is no transition to chaos by using the connectivity strength of the weights g as a bifurcation parameter in non-ReLU threshold power-law RNNs. For any finite N and $\boldsymbol{\omega}$, if the system displays chaotic dynamics for some g, it does so for all $g > 0$. Indeed, in [7], the authors remark that the chaos found in the threshold-power-law networks for $k = \frac{1}{2}$ can persist for very small g (see Figure 2a in [7]) although we remark that the network in [7] differs from the RNNs considered here as the authors consider external inputs rather than completely autonomous dynamics. Despite the (very) simple derivation of Lemma 1, it does not appear to be explicitly mentioned in the literature surrounding threshold power-law RNNs to the best of our knowledge.

Further, we remark that Lemma 1 also has implications for reservoirs where the firing rates saturate due to refractory periods [15]. The firing rate for these networks is given by

$$f_\tau(z) = \frac{f(z)}{\tau f(z) + 1} \tag{10}$$

which asymptotically converges to $f_\tau(z) \to \frac{1}{\tau}$ as $z \to \infty$ and $f_\tau(z) \to f(z)$ as $z \to 0$. Consider the threshold power-law RNN with a refractory period given by

$$\frac{d\boldsymbol{z}}{dt} = -\boldsymbol{z} + g_\tau \boldsymbol{\omega} f_\tau(\boldsymbol{z}) \tag{11}$$

coupled with coupling parameter g_τ, where $f(z)$ is a threshold power-law transfer function as in Equation (2) while $f_\tau(z)$ is given by Equation (10). In the limit that $g_\tau \to 0$, the solution(s) to equation (11) can be shown to converge to rescaled solutions of

$$\frac{d\boldsymbol{y}}{dt} = -\boldsymbol{y} + \boldsymbol{\omega} f(\boldsymbol{y}). \tag{12}$$

This was confirmed numerically in Figure 2 for chaotic (Figure 2B), oscillatory (Figure 2C), and equilibria solutions (Figure 2D).

The proof of Theorem 1 follows immediately from Lemma 1 by considering the trained reservoir:

$$\frac{d\boldsymbol{z}}{dt} = -\boldsymbol{z} + (g^*\boldsymbol{\omega} + \boldsymbol{\eta}\boldsymbol{\phi}^T) f(\boldsymbol{z}), \quad \hat{\boldsymbol{x}} = \boldsymbol{\phi}^T \boldsymbol{f}(\boldsymbol{z}) \tag{13}$$

which is equivalent to the rescaled system

$$\frac{d\boldsymbol{y}}{dt} = -\boldsymbol{y} + (g\boldsymbol{\omega} + \hat{\boldsymbol{\eta}}\hat{\boldsymbol{\phi}}^T) f(\boldsymbol{y}), \quad \hat{\boldsymbol{x}} = \hat{\boldsymbol{\phi}}^T f(\boldsymbol{y}) \tag{14}$$

where

$$\hat{\boldsymbol{\phi}} = \boldsymbol{\phi} \left(\frac{g}{g^*}\right)^{\frac{k}{k-1}}, \quad \hat{\boldsymbol{\eta}} = \boldsymbol{\eta} \left(\frac{g^*}{g}\right)^{\frac{1}{k-1}} \tag{15}$$

Thus, if a threshold power-law reservoir can be trained for a single coupling strength g (for a particular supervisor), it can be trained for all coupling strengths.

Next, we sought to determine if these threshold power-law RNNs could be trained at all, that is if the chaos could be tamed onto a target dynamical system. We found that threshold power-law RNNs (for $k = 1/2$) could indeed learn different dynamical systems with FORCE training (Figure 3) [9]. We first considered training the networks to mimic multi-frequency oscillations produced from randomly generated oscillators of the form

$$x(t) = \sum_{j=1}^{n_o} a_j \cos(2\pi t f_j) \tag{16}$$

where a_j is a randomly generated amplitude drawn from a standard normal, and f_j is a fixed frequency (Materials and Methods). We found that networks could be trained for $k = \frac{1}{2}$ (Figure 3A-B), and with other powers for $k < 1$ (Figure 3C-D). We found that using $k > 1$ led either to unstable system dynamics (blow-up, not shown) or convergence to non-firing dynamics (not shown). Further, we found that the test error in training randomly generated oscillatory supervisors decreased steadily with power k (Figure 4) until $k \to 1$, where the reservoirs could lose stability or testing accuracy. Thus, threshold power-law RNNs can be trained with FORCE reliably for sufficiently sublinear powers for the randomly generated oscillatory supervisors considered here.

Next, we tested threshold power-law reservoirs on accurate chaotic time series predictions (Figure 5(B-F) with the Rossler system. We successfully trained a threshold power-law RNN with FORCE training to mimic the Rossler system (Figure 5A-F) with a short period of training. Further, as confirmed by theorem 1, the trained reservoir for one value of the coupling strength ($g^* = \frac{1.1}{\sqrt{N}}$) could be rescaled to a different value of the coupling strength ($g = \frac{1.9}{\sqrt{N}}$) with equation (15) rescaling the encoders and decoders.

Collectively, these results analytically and numerically confirm that the coupling strength g acts as a scale parameter for threshold power-law RNNs. This implies that when used as a reservoir, the coupling strength, g, does not dictate the trainability of these types of reservoirs for any supervisor.

Discussion

Through both analytical derivations and numerical simulations, we have shown that threshold power-law RNNs, both trained and untrained, operate differently from their sigmoidal counterparts [1]. In particular, the overall coupling strength of these networks does not influence the chaotic dynamics of the system, and as a result, does not influence the accuracy of any trained network. The coupling strength scales solutions up or down in amplitude. The scaling invariance in threshold power-law RNNs even applies to networks where the firing rate has a refractory period, as all weak coupling solutions $g_\tau \to 0$ are scaled solutions of the threshold power-law RNN where the firing rates have no refractory periods.

We remark that the fact that g acts as a scale-parameter was partially considered in [6] and [7]. In particular, both studies show that for large network limits, chaotic solutions emerge for all $g \neq 0$. Here, we show that this result is more general: for any finite N, the solutions that exist for some $g = g^*$ exist for all $g > 0$, chaotic or otherwise, as the coupling strength does not influence the qualitative behaviours of threshold power-law RNNs. Similarly, if the non-firing equilibrium point ($\boldsymbol{z} = 0$) is unstable for any finite N for the weight matrix $g^*\boldsymbol{\omega}$, then it is unstable for all coupling strengths with the same $\boldsymbol{\omega}$. The dynamic mean-field theories considered by [6] and [7] guarantee the existence of high-dimensional chaotic solutions based on excitatory/inhibitory balance, similar to those of classical sigmoidal networks in [1] as $N \to \infty$. The work here shows that the immediate onset to chaos does not impact the ability of these networks to be trained with the techniques of reservoir computing, and that the asymptotic derivations for large N are caused by the coupling strength acting as a scale parameter. We remark that threshold power-law neural networks have also been considered previously for certain values of k under other conditions [29–32].

While we have demonstrated that the parameter g does not directly influence the loss of a trained reservoir or the dynamics of an untrained initial reservoir, this does not imply that the g parameter is not important in the context of explicitly training an RNN. First, we remark that Theorem 1 states that the learned low-rank perturbation for $g = g^*$ can be transformed so that the initial reservoir had a coupling strength of g, but this does not imply that a learning algorithm will necessarily learn the appropriate scaling for an encoder/decoder pair. While g does not impact the dynamics of the reservoir qualitatively, this scaling is still practically important for training. If g is too small, and the learning algorithm applied does not somehow explicitly scale the encoder/decoder pair to account for g, then a functional low-rank perturbation that stabilizes the dynamics of the reservoir to $\hat{\boldsymbol{x}} \approx \boldsymbol{x}(t)$ may not be found.

Collectively, this work demonstrates that threshold power-law RNNS, which in some ways are more biologically plausible than sigmoidal RNNs, have very different qualitative behaviours. The chaos that emerges for large networks can either always be tamed for any coupling strength, or can never be tamed.

Acknowledgments

We thank Claudia Clopath for her insightful comments. This work was funded by an NSERC Discovery Grant (RGPIN/04568-2020), a Canada Research Chair (CRC-2019-00416), and the Hotchkiss Brain Institute.

Materials and Methods

FORCE Training

Chaotic RNNs were trained with the FORCE algorithm [9, 10, 27]. In FORCE training, an initially chaotic RNN given by

$$\frac{d\boldsymbol{z}}{dt} = -\boldsymbol{z} + g\boldsymbol{\omega} f(\boldsymbol{z}) \tag{17}$$

is stabilized onto an m-dimensional supervisor $\boldsymbol{x}(t)$ with a linear decoder, $\boldsymbol{\phi}$ where $\hat{\boldsymbol{x}(t)} = \boldsymbol{\phi}^T f(\boldsymbol{z}(t))$, and $\hat{x}(t) \approx x(t)$. After training, the system equations are given by:

$$\frac{d\boldsymbol{z}}{dt} = -\boldsymbol{z} + (g\boldsymbol{\omega} + \boldsymbol{\eta}\boldsymbol{\phi}) f(\boldsymbol{z}) \tag{18}$$

The encoder matrix was an $N \times k$ matrix where each element was drawn from a uniform random distribution defined on $[-1, 1]$. The decoder was updated with Recursive Least Squares (RLS)

$$\boldsymbol{P}_{n+1} = \boldsymbol{P}_n - \frac{\boldsymbol{r}_n^T \boldsymbol{P}_n^T \boldsymbol{P}_n \boldsymbol{r}_n}{1 + \boldsymbol{r}_n^T \boldsymbol{P}_n \boldsymbol{r}_n} \tag{19}$$

$$\boldsymbol{\phi}_{n+1} = \boldsymbol{\phi}_n - \frac{\boldsymbol{P}_n \boldsymbol{e}_n^T}{1 + \boldsymbol{r}_n^T \boldsymbol{P}_n \boldsymbol{r}_n} \tag{20}$$

where $e_n = \hat{x}_n - x_n$. The index n corresponds to every 3 time steps in the numerical integration here. All networks considered were integrated with a forward Euler scheme with a time step of $\Delta = 10^{-2}$ in MATLAB 2023a.

Supervisors

The random oscillatory supervisors (Figure 2, 3 were generated as

$$x(t) = \sum_{j=1}^{3} a_j \cos(2\pi t f_j) \tag{21}$$

where $f_1 = 1/6$, $f_2 = 1/8$ and $f_3 = 1/10$. The Rossler system [33] was given by

$$\frac{dx_1(t)}{dt} = -x_2(t) - x_3(t) \tag{22}$$

$$\frac{dx_2(t)}{dt} = x_1(t) + a x_2(t) \tag{23}$$

$$\frac{dx_3(t)}{dt} = b + x_3(t)(x_1(t) - c) \tag{24}$$

where $a = 0.2, b = 0.2, c = 5.7$ and $x_1(t), x_2(t)$ and $x_3(t)$ were z-transformed (mean 0, standard deviation 1) before being used as a supervisor.

Figures

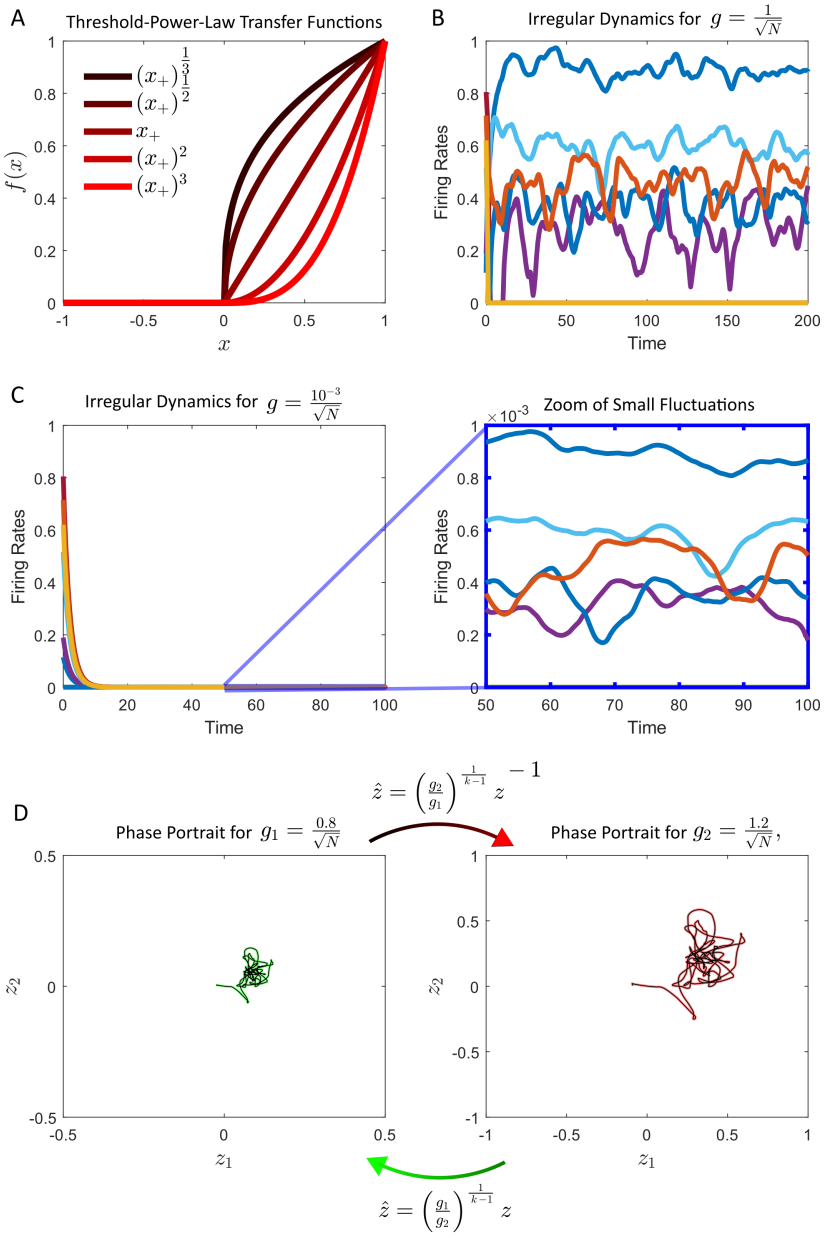

Figure 1: Threshold Power-Law Recurrent Neural Networks. **(A)** Various transfer functions for threshold power-law neural networks, where $x_+ = \max x, 0$. **(B)** A simulation of a threshold power-law RNN with $N = 1000$ neurons, and $k = \frac{1}{2}$, where ω_{ij} is drawn from a standard normal distribution with a coupling strength of $g = \sqrt{N}^{-1}$. The system displays irregular dynamics indicative of a chaotic solution as predicted from dynamic mean field theoreis [6, 7] for large N. **(C)** Decreasing g to $10^{-3}\sqrt{N}^{-1}$ leads to an initial decay to a neighbourhood near the origin (left), but a zoom (right) reveals small firing rate fluctuations. All other parameters are as in (B). **(D)** Lemma 1 shows that solutions for one value of g are rescaled solutions for any other value of g. This is shown numerically with $g_1 = 0.8\sqrt{N}^{-1}$ and $g_2 = 1.2\sqrt{N}^{-1}$ with the transform defined in the red and green arrows to rescale the respective solutions for the two values of g. The initial conditions are also rescaled. All weights were drawn from a standard normal distribution.

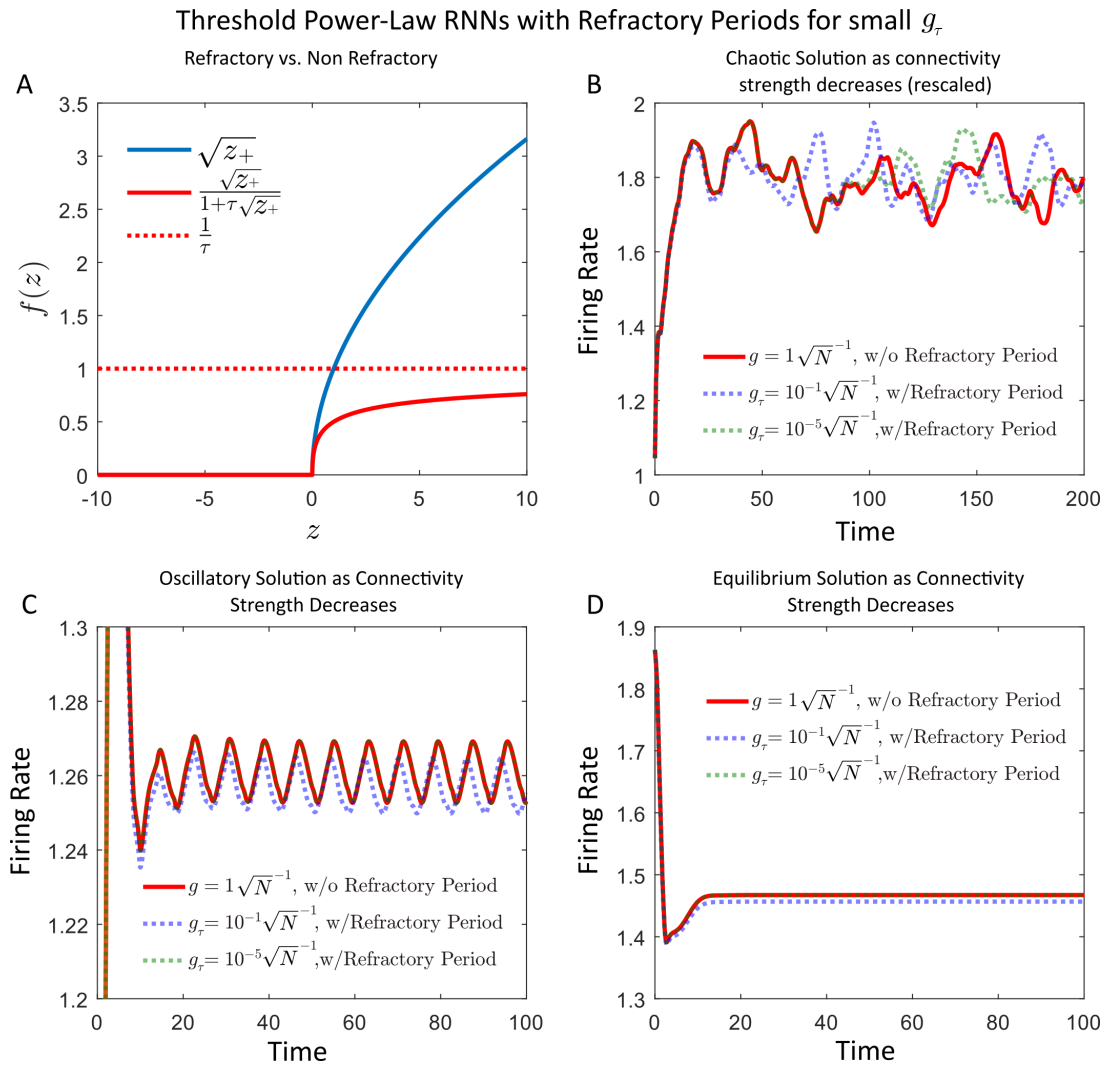

Figure 2: Threshold Power-Law RNNs with Refractory Periods. **(A)** The firing rates for a threshold power-law RNN with $k = \frac{1}{2}$ (blue) and with a refractory period (red). The firing with a refractory period τ converges to τ^{-1} for high input currents (z). **(B)** The firing rate for a simulated network without a refractory period (red, $N = 2000$, $g = \sqrt{N}^{-1}$ and with a refractory period for $g_\tau = 10^{-1}\sqrt{N}^{-1}$ (blue-dashed) and $g_\tau = 10^{-5}\sqrt{N}^{-1}$ (green-dashed). As $g_\tau \to 0$, the solutions of the network with a refractory period converge to solutions of the network without one. Note that the green and blue solutions were re-scaled with $\hat{z}(t) = z(t)(\frac{g_\tau}{g})^{1/(k-1)}$ where g_τ denotes the network coupling strength for the network with a refractory period τ. **(C)** Identical as in (B) only with oscillatory solutions as $g_\tau \to 0$ by with $N = 50$ neurons. **(D)** Identical as in (C), only with an equilibrium point solution as $g_\tau \to 0$. All weights were drawn from a standard normal distribution.

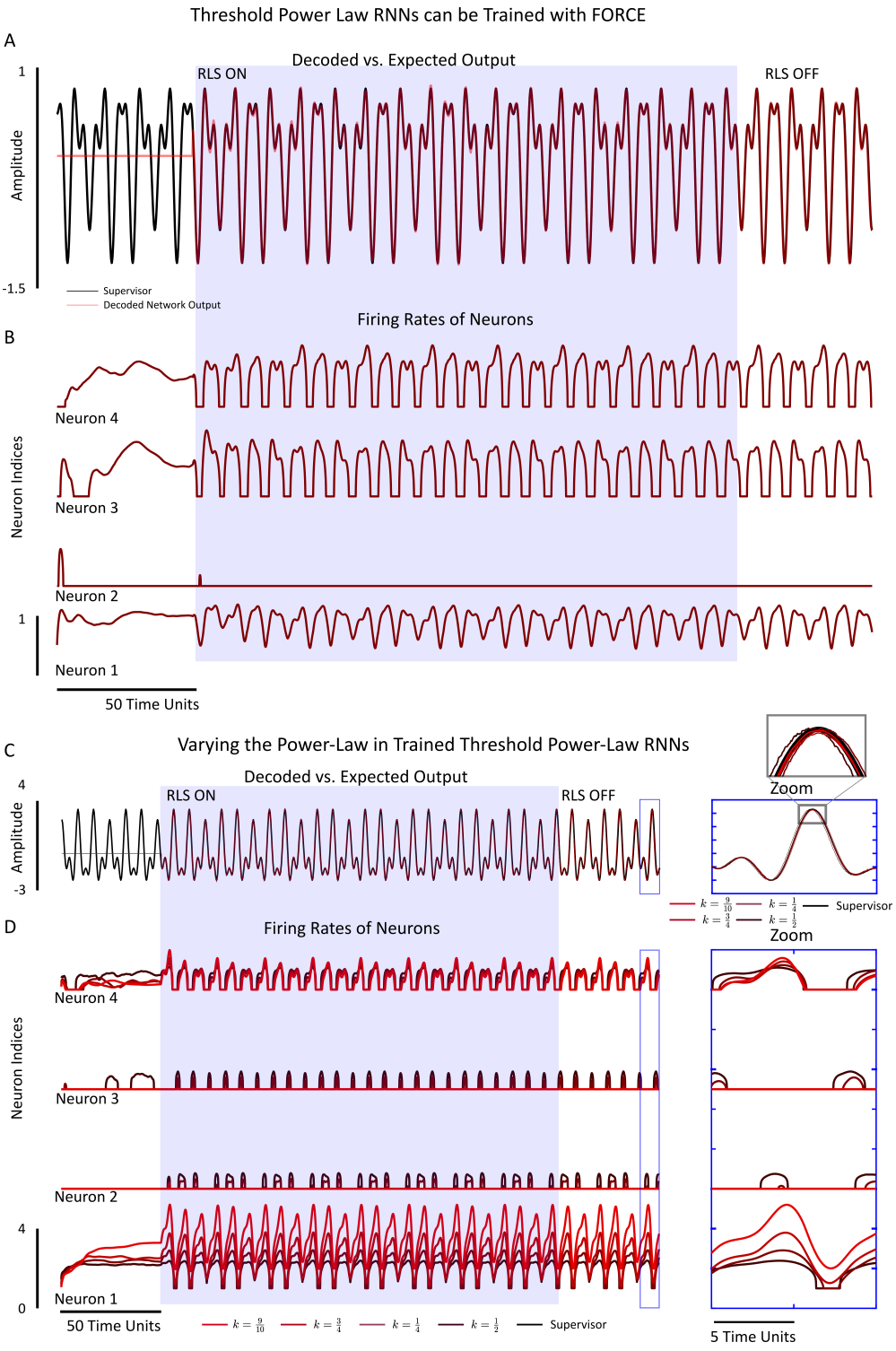

Figure 3: Threshold Power-Law RNNs can be Trained with FORCE. **(A)** The supervisor (black), a random sum of oscillators (Materials and Methods) vs the network approximation (red). RLS was turned on after 50 time units and turned off for the last 50 time units in a 300 time unit simulation. The network consisted of $N = 2000$ neurons with $k = \frac{1}{2}$ and $g = 1$. **(B)** The firing rates for 4 neurons, time aligned with (A). **(C)** The supervisor (as in (A), but with a different random sum, black) vs networks with increasingl larger powers k (reds). RLS was turned on after 50 time units and off for the last 50 time units in a 300 time unit simulation. A zoom of the last 10 time units is shown on the right. **(D)** The firing rates for four neurons, time-aligned with (C). Note that all 4 networks used the same initial state and the same initial reservoir weight matrix with $g = 1.5\sqrt{N}^{-1}$ in all cases. A zoom of the last 10 time units is shown on the right. All weights were drawn from a standard normal distribution.

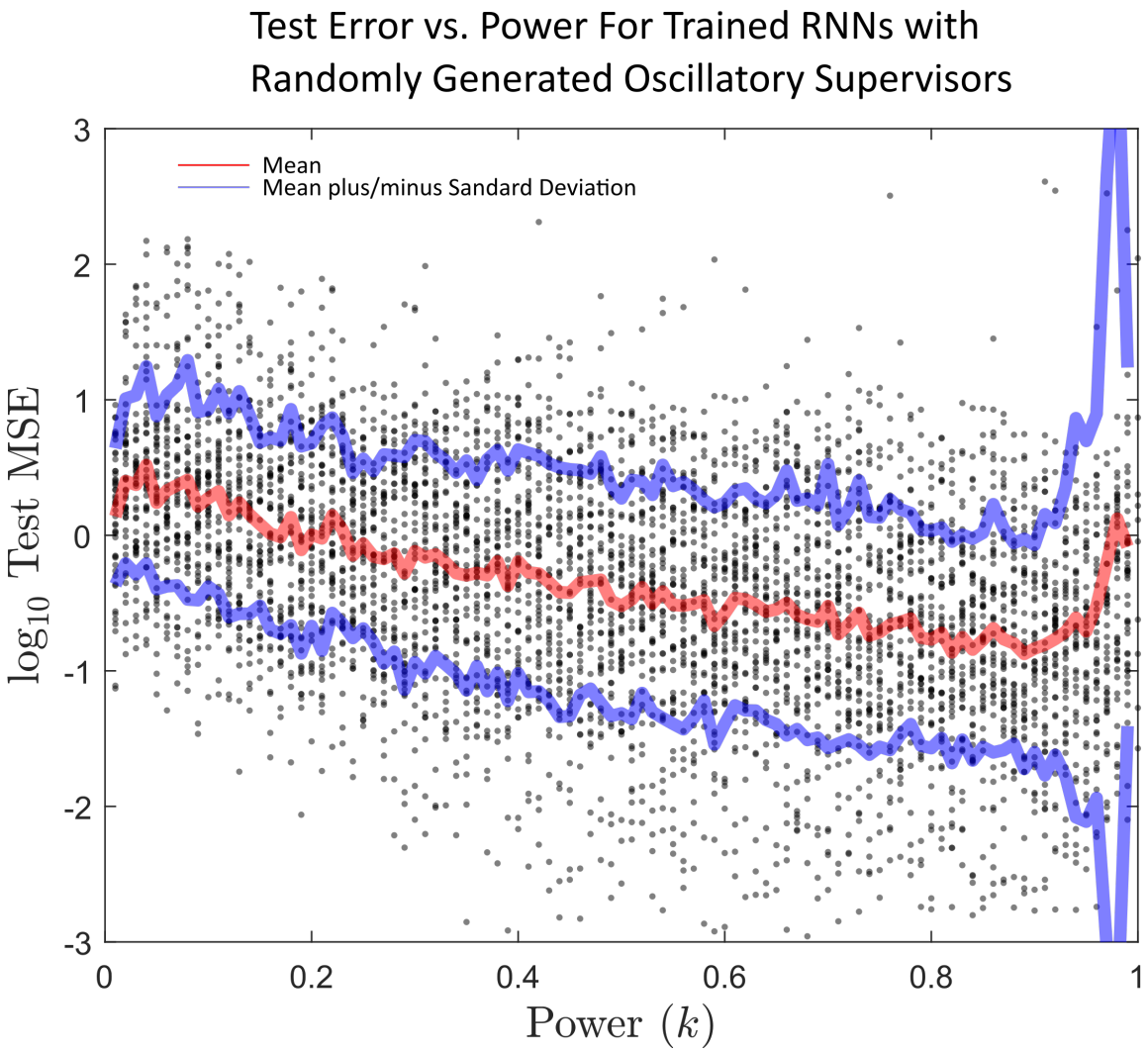

Figure 4: Test Error vs. Power in Trained Threshold Power-Law RNNs. The power k varied from $1/100$ to $100/100$ in increments of $1/100$. For each value of k, 50 random networks consisting of $N = 2000$ neurons were generated and FORCE trained with randomly generated oscillatory supervisors as in Figure 3. The log of the test MSE is shown as individual points (black dots) with the mean (red) and mean $\pm$ standard deviation in blue. The test error monotonically decreases until $k \approx 1$ where the networks start becoming unstable. Each network was simulated for 500 time units with RLS turned on after the first 50 time units and off for the last 50 time units (testing). All networks used a constant $g = 1.5\sqrt{N}^{-1}$. All weights were drawn from a standard normal distribution.

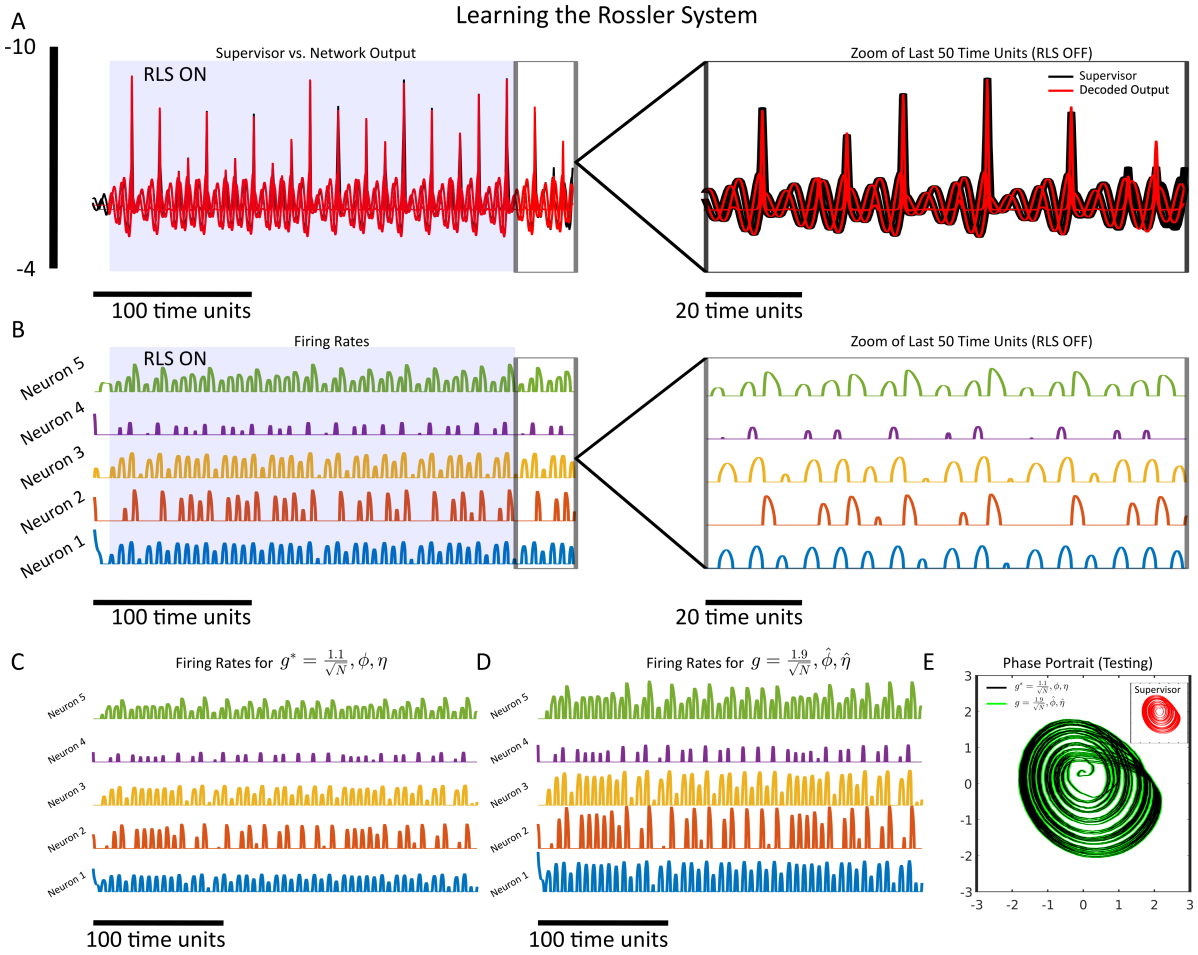

Figure 5: Trained Threshold-Power-Law Recurrent Neural Networks can be Rescaled to Any Reservoir Strength. **(A)** A network of 2000 neurons with $g = \frac{1.1}{\sqrt{N}}$ was trained on a complex oscillator task with FORCE training. RLS was turned on on 50 time units and turned off at 200 time units. The decoded output (red) vs the target supervisor (black) is shown on top, while the firing rates for 5 neurons are shown on the bottom. **(B)** A network of 2000 neurons with $g = \frac{1.1}{\sqrt{N}}$ was trained to learn the Rossler dynamical system with FORCE training (network, red, system, Rossler, black). A zoom of the last 50 time units with RLS off is shown on the right, while the full simulation is shown on on the left. **(C)** The firing rates for 5 randomly selected neuron. The zoom on the right is aligned with the zoom in (B). **(D)** The firing rates for the reservoir with $g^* = \frac{1.1}{\sqrt{N}}$ during testing. **(E)** The firing rates for the reservoir with $g = \frac{1.0}{\sqrt{N}}$ during testing using the encoder and decoder pair $\hat{\boldsymbol{\eta}}$ and $\hat{\boldsymbol{\phi}}$ where ϕ as described in the text. **(F)** The phase portrait of the x_1 vs x_2 variable for the network trained with g^* (black) and g (green). The Rossler inset is shown in red.

References

[1] Carl Van Vreeswijk and Haim Sompolinsky. Chaos in neuronal networks with balanced excitatory and inhibitory activity. *Science*, 274(5293):1724–1726, 1996.

[2] Rainer Engelken, Alessandro Ingrosso, Ramin Khajeh, Sven Goedeke, and LF Abbott. Input correlations impede suppression of chaos and learning in balanced firing-rate networks. *PLOS Computational Biology*, 18(12):e1010590, 2022.

[3] Johnatan Aljadeff, Merav Stern, and Tatyana Sharpee. Transition to chaos in random networks with cell-type-specific connectivity. *Physical review letters*, 114(8):088101, 2015.

[4] Merav Stern, Nicolae Istrate, and Luca Mazzucato. A reservoir of timescales emerges in recurrent circuits with heterogeneous neural assemblies. *Elife*, 12:e86552, 2023.

[5] Jordan M Culp and Wilten Nicola. Input driven synchronization of chaotic neural networks with analyticaly determined conditional lyapunov exponents. *Nonlinear Dynamics*, 113(10):12131–12141, 2025.

[6] Omri Harish and David Hansel. Asynchronous rate chaos in spiking neuronal circuits. *PLoS computational biology*, 11(7):e1004266, 2015.

[7] Jonathan Kadmon and Haim Sompolinsky. Transition to chaos in random neuronal networks. *Physical Review X*, 5(4):041030, 2015.

[8] Andrea Mattera, Valerio Alfieri, Giovanni Granato, and Gianluca Baldassarre. Chaotic recurrent neural networks for brain modelling: A review. *Neural Networks*, 184:107079, 2025.

[9] David Sussillo and Larry F Abbott. Generating coherent patterns of activity from chaotic neural networks. *Neuron*, 63(4):544–557, 2009.

[10] Brian DePasquale, Christopher J Cueva, Kanaka Rajan, G Sean Escola, and LF Abbott. full-force: A target-based method for training recurrent networks. *PloS one*, 13(2):e0191527, 2018.

[11] Toshitake Asabuki and Claudia Clopath. Taming the chaos gently: a predictive alignment learning rule in recurrent neural networks. *Nature Communications*, 16(1):6784, 2025.

[12] Denis Turcu and LF Abbott. Sparse rnns can support high-capacity classification. *PLOS Computational Biology*, 18(12):e1010759, 2022.

[13] Herbert Jaeger. Echo state network. *scholarpedia*, 2(9):2330, 2007.

[14] Min Yan, Can Huang, Peter Bienstman, Peter Tino, Wei Lin, and Jie Sun. Emerging opportunities and challenges for the future of reservoir computing. *Nature Communications*, 15(1):2056, 2024.

[15] Bard Ermentrout and David Hillel Terman. *Mathematical foundations of neuroscience*, volume 35. Springer, 2010.

[16] Pierre Ekelmans, Nataliya Kraynyukova, and Tatjana Tchumatchenko. Targeting operational regimes of interest in recurrent neural networks. *PLOS Computational Biology*, 19(5):e1011097, 2023.

[17] Nataliya Kraynyukova and Tatjana Tchumatchenko. Stabilized supralinear network can give rise to bistable, oscillatory, and persistent activity. *Proceedings of the National Academy of Sciences*, 115(13):3464–3469, 2018.

[18] Daniel B Rubin, Stephen D Van Hooser, and Kenneth D Miller. The stabilized supralinear network: a unifying circuit motif underlying multi-input integration in sensory cortex. *Neuron*, 85(2):402–417, 2015.

[19] Yashar Ahmadian, Daniel B Rubin, and Kenneth D Miller. Analysis of the stabilized supralinear network. *Neural computation*, 25(8):1994–2037, 2013.

[20] Bard Ermentrout. Linearization of fi curves by adaptation. *Neural computation*, 10(7):1721–1729, 1998.

[21] Rainer Engelken, Fred Wolf, and Larry F Abbott. Lyapunov spectra of chaotic recurrent neural networks. *Physical Review Research*, 5(4):043044, 2023.

[22] Chengcheng Huang and Brent Doiron. Once upon a (slow) time in the land of recurrent neuronal networks.... *Current opinion in neurobiology*, 46:31–38, 2017.

[23] Rainer Engelken and Sven Goedeke. A time-resolved theory of information encoding in recurrent neural networks. *Advances in Neural Information Processing Systems*, 35:35490–35503, 2022.

[24] Guillaume Hennequin and Máté Lengyel. Characterizing variability in nonlinear recurrent neuronal networks. *arXiv preprint arXiv:1610.03110*, 2016.

[25] Alexander van Meegen, Tobias Kühn, and Moritz Helias. Large-deviation approach to random recurrent neuronal networks: Parameter inference and fluctuation-induced transitions. *Physical review letters*, 127(15):158302, 2021.

[26] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

[27] Wilten Nicola and Claudia Clopath. Supervised learning in spiking neural networks with force training. *Nature communications*, 8(1):2208, 2017.

[28] Mantas Lukoševičius. A practical guide to applying echo state networks. In *Neural Networks: Tricks of the Trade: Second Edition*, pages 659–686. Springer, 2012.

[29] Behnam Neyshabur, Yuhuai Wu, Russ R Salakhutdinov, and Nati Srebro. Path-normalized optimization of recurrent neural networks with relu activations. *Advances in Neural Information Processing Systems*, 29, 2016.

[30] Sachin S Talathi and Aniket Vartak. Improving performance of recurrent neural network with relu nonlinearity. *arXiv preprint arXiv:1511.03771*, 2015.

[31] Mohammadreza Soltani and Chinmay Hegde. Fast and provable algorithms for learning two-layer polynomial neural networks. *IEEE Transactions on Signal Processing*, 67(13):3361–3371, 2019.

[32] Wilten Nicola, Bryan Tripp, and Matthew Scott. Obtaining arbitrary prescribed mean field dynamics for recurrently coupled networks of type-i spiking neurons with analytically determined weights. *Frontiers in Computational Neuroscience*, 10:15, 2016.

[33] Otto E Rössler. An equation for continuous chaos. *Physics Letters A*, 57(5):397–398, 1976.