An homeostatic activity-dependent structural plasticity algorithm for richer input combination

Tanguy Cazalets, Joni Dambre

***Abstract*—This paper introduces a novel rate-based variant of homeostatic activity-dependent structural plasticity (HADSP) for echo state networks. Despite its importance in brain development, structural plasticity has been largely overlooked in artificial neural networks. Our algorithm, although using only homeostatic plasticity, let emerge principles of Hebbian learning. Our analysis sheds light on the information processing capabilities of HADSP-powered echo state networks and suggests that HADSP effectively leverages the inter-relationships of the network's inputs. The study highlights the potential for rate-based HADSP to contribute to the field of computational neuroscience and plasticity in echo state networks. Furthermore, our findings highlight the crucial role of structural plasticity in influencing network function and organization and contribute significantly to the ongoing research on leveraging plasticity for the advancement of reservoir computing techniques.**

I. Introduction

Homeostatic activity-dependent structural plasticity (HADSP) is the ability of a network to change its structural organization in response to a pattern of activity.

Structural plasticity refers to the brain's ability to change its connectivity through the formation or elimination of synapses, which in the context of artificial neural networks translates into changing a connection weight from 0 to a non-zero value or vice versa. Structural plasticity is a fundamental mechanism in the formation and maturation of biological neural circuits during development [1] [2]. In the field of neuroscience, the study of how structural plasticity contributes to the development of neural circuits has been the subject of significant scientific research [3]. It is known to happen so that neurons maintain their homeostasis [2] and to be guided by electrical activity [4]. Structural plasticity is particularly puzzling in the context of artificial neural networks because it is completely absent. Indeed, in the context of reservoir computing and deep learning, the architecture of the neural network is usually fixed, and learning occurs only through changes in the strength of pre-existing connections.

This is particularly striking in the context of echo state networks (ESNs), the rate-based version of reservoir computing. In ESNs, connections between neurons are generated randomly, an approach that has been described as "the antithesis of the 'optimal'" approach [5]. And indeed, the use of random weights is in stark contrast to the traditional approach of optimizing connections for maximum performance. As such, the study of reservoir computing presents an important opportunity for research in plasticity, and recent studies have explored this avenue by investigating the effects of various non-structural plasticity rules on reservoir computation performance [6] [7] [8]. Compared to previous studies, our work focuses on a structural plasticity rule.

This project has received funding from the European Union's Horizon 2020 research and innovation program under the Marie Skłodowska-Curie grant agreement No 860949

Hebbian learning and homeostatic plasticity are two fundamental mechanisms of neural plasticity that alter the strength of connections between neurons based on their activity. Hebbian plasticity focuses on extracting correlations from sensory information, while homeostatic plasticity maintains overall stability by balancing activity. The two mechanisms are complementary [9] and contribute to the adaptiveness of neural systems, yet the relationship between them is not fully understood. Recent studies have shown that Hebbian plasticity can emerge from homeostatic plasticity in spiking neural networks [4] [10]. We build on this work to create a rate-based HADSP algorithm and apply it in the context of echo state networks.

This paper first presents the necessary background (section II) for echo state networks and describes the rate-based HADSP algorithm. We also introduce the main elements of our experiment and the techniques used in this section. Section III then offers an in-depth analysis of the algorithm and in particular the different types of network configurations obtained from HADSP. We show that HADSP leads to different computational properties compared to a randomly instantiated reservoir using the information processing capacity. In section IV we show that our algorithm also implements a form of Hebbian learning and test our algorithm on the Mackey-Glass equations benchmark to demonstrate the capabilities of the network to exploit the redundancy in its inputs to improve performance on this benchmark.

This study drew inspiration from the field of computational neuroscience, and as such, we use the terms state, activation, activity and firing rate of neurons as synonymous.

II. Background

A. Echo state networks

An illustration of the Echo state networks framework is given in Fig. 1. It consists of n neurons. The evolution of the vector states $\mathbf{x}$ of the n neurons is determined by the interactions between the connection matrix $\mathbf{W}$, the input matrix $\mathbf{W_{in}}$, the input $u[t]$, the bias vector $\mathbf{b}$ and the activation function σ, combined in the following equation :

$$\mathbf{x}[t+1] = \sigma(\mathbf{W} \times \mathbf{x}[t] + \mathbf{W_{in}} \times u[t]) + \mathbf{b} \tag{1}$$

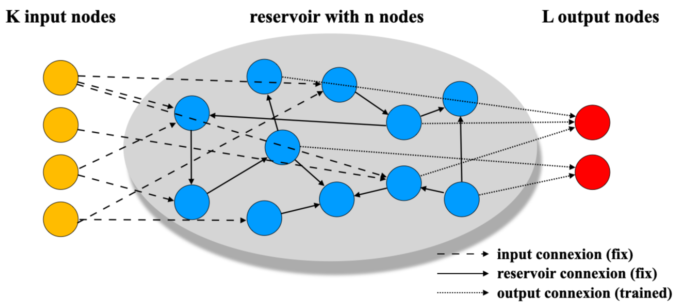

Fig. 1. Schematic architecture of reservoir computing

We take σ as the hyperbolic tangent function. The purpose of the bias term $\mathbf{b}$ is to enrich the dynamics of the network, it is however kept low enough to not be preponderant. For this paper, $\mathbf{b}$ is always a $1 \times n$ vector, sampled randomly from a normal distribution of mean 0.1 and standard deviation 0.1. The dimensions of the other matrices are $n \times n$ for $\mathbf{W}$ and $1 \times n$ for $\mathbf{W_{in}}$.

Once $\mathbf{W}$, $\mathbf{W_{in}}$ and $\mathbf{b}$ are fixed, the desired output is obtain through the following equation :

$$y[t+1] = \mathbf{W_{out}} \times \mathbf{x}[t+1] + \mathbf{b_{out}} \tag{2}$$

where $\mathbf{W_{out}}$, $\mathbf{b_{out}}$ are learned using ridge regression such that the generated train output, $y[t]$, closely approximates a desired target output, $y^{target}[t]$ [5].

B. Presentation of the HADSP algorithm

1) Growing rule: This rule has been derived from a computational neuroscience context with a spiking neural network. In biology, the calcium concentration in the soma of neurons is thought to be proportional to the firing rate and to potentially impact the growth of dendrites and axons [11] [10]. In this model, neurons have multiple synapses of fixed weight. Similarly, in our algorithm, we consider that neurons can have multiple connections between them. A connection coming from node i to node j means that the value $w_{ij} = 0.05$. However, the same pair of neurons can have multiple synapses; thus if node j has k connections coming from node i, $w_{ij} = k * 0.05$. Finally, self-connections are not allowed in our model.

We limit the maximum number of partners γ a neuron can have, meaning that we always have $k \leq \gamma$.

The value of a 0.05 weight increment is linked to the input strength. Changing this value influences the regime's dynamics. This parameter should be kept low enough to allow the HADSP to recombine the inputs without creating an unstable regime. In the context of our experiments, a weight increment value of 0.05 was found to be optimal. Every 100 steps, a growth indicator, $\mathbf{\Delta z}$, is calculated for each neuron according to the following formula :

$$\mathbf{\Delta z} = -\frac{1}{\beta}(\langle \mathbf{x} \rangle - \rho) \tag{3}$$

where $\langle \mathbf{x} \rangle$ is the average of the state vector $\mathbf{x}$ over the increment period. $\mathbf{\Delta z}$ depends on two hyperparameters : the growth parameter β and the target rate ρ.

Based on the resulting value of Δz_j of the j-th neuron, one input connection is added if $\Delta z_j < -1$ or pruned if $\Delta z_j > +1$.

Thus, when a neuron's activity falls below the -1 threshold, a new connection is established to increase its activity. The creation of new connections is restricted to neurons that have been identified as requiring additional connections. Among all those possible partners, one is selected randomly to create a new connection. Similarly, when a neuron's activity exceeds the +1 threshold, an existing connection is pruned in order to decrease its activity. Note that the pruning of connections is performed independently of the state of the neuron's partners, regardless of whether they also need to decrease their activity or not. In other words, it does not take into account the relation between the activities of the neurons pairs, which differentiates it from Hebbian mechanisms.

2) On the need of excitatory neurons: The algorithm is not effective in the presence of mixed inhibitory and excitatory neurons. Indeed, since it aims to regulate the firing rate of neurons through the creation and pruning of connections, it requires a clear understanding of the effect of input on neuron activity.

In order to increase the firing rate of a neuron, it is necessary to determine whether the input will have a positive or negative impact on the rate. The easiest option to ensure this is to make sure previous neurons have strictly positive or negative output to properly guide the process.

Consequently, we utilize only excitatory neurons, this choice limits the computational properties of the ESN, in order to maintain the desired level of interpretability and functionality in the algorithm.

C. Experiments

1) Important characteristics of the reservoir: Key characteristics to the reservoir behavior include the spectral radius of $\mathbf{W}$ and the scaling of $\mathbf{W_{in}}$ [5].

The *spectral radius* is the largest eigenvalue λ_i of the weight matrix :

$$sr(\mathbf{W}) = \max_{i=1}^{n} |\lambda_i| \tag{4}$$

It influences the rate at which information decays or amplifies in the reservoir over time. Proper selection of the spectral radius ensures that the reservoir is capable of producing good output predictions. In practice, $sr(\mathbf{W})$ is set to maximize performance on a task, usually to a value around 1.

Input scaling of the weight matrix $\mathbf{W_{in}}$ is important in echo state networks because it affects the dynamic range of inputs that are fed into the network. A properly scaled input ensures that the inputs are within a range that allows the network to capture the relevant information from the input signal. If the input is not scaled properly, then the network may become saturated, leading to distorted and inaccurate output.

In our context, input scaling is twice as important because if the magnitude of the inputs is too high, the network activation will already be too high and our algorithm will not be able to combine them to reach the desired target activation. To define the scaling of the inputs, we try to keep the perceived input at a maximum amplitude, that is 3 times lower than the target rate of our HADSP algorithm.

2) Information Processing Capacity: A first characterization of information processing within our networks is through the information processing capacity metric (IPC) [12]. The IPC generalizes the linear memory capacity [13], and serves as a quantitative measure of the network's capacity for non-linear computation and memory depth. For a network $\mathbf{W}$ whose input is sampled from a uniform random distribution in the range [-1,1], the capacity to reproduce one function z is measured by :

$$CT_{X,\frac{z}{2}} = 1 - \frac{\min_{\mathbf{W}} MSE_T[\hat{z}]}{\langle z^2 \rangle}_T \tag{5}$$

where $\langle z^2 \rangle_T = \frac{1}{T}\sum_{t=1}^{T} z^2[t]$

The IPC is calculated by summing the capacities of a projection of the input onto an orthonormal set of functions.

With this tool, our aim is to assess if and how the network dynamics of a network that has been formed with HADSP are different from a network whose weights are randomly sampled from a uniform distribution.

3) Mackey-Glass benchmark: The IPC offers a task-agnostic analysis of the computations a network performs on uniform independent and identically distributed input sequences. In a second experiment, we evaluated how HADSP adapted the network to non-independent and non-identically distributed inputs and whether this adaptation could be beneficial from a task-based perspective. For this analysis, we used a benchmark that is consistent with those found in existing literature [8] [14] even though direct comparison remains challenging due to our model relying on excitatory connections only.

We employ the Mackey-Glass series prediction task, a widely utilized benchmark dataset derived from a time-delay differential equation to evaluate the performance of the HADSP generated network [15] [16]. The Mackey-Glass attractor equations describe the dynamics of an endocrine system and predict the concentration of a hormone in the bloodstream over time :

$$\frac{du}{dt} = \beta \frac{u(t-\tau)}{1+u^{10}(t-\tau)} - \gamma u(t) \tag{6}$$

where $u(t)$ is the state of the system at time t, β and γ are positive constants, and τ is the time delay. τ can be adjusted to leverage the behavior of the system. We evaluated the prediction performance for two values of the delay: one resulting in mild chaotic behavior (τ = 17) and another in strong chaotic behavior (τ = 30).

The task for this reservoir is k-step ahead prediction and performance is evaluated using the Normalized Root Mean Square Error (NRMSE), a commonly used metric for evaluating the accuracy of a model's prediction. Mathematically, it is defined as:

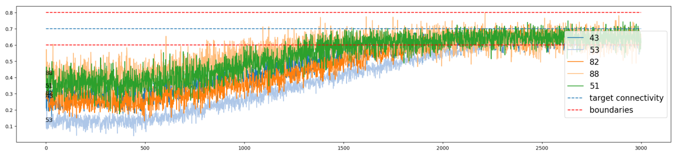

Fig. 2. Evolution of the activity upon the effects of the activity-dependant homeostatic plasticity for 5 randomly selected neurons. The blue dotted line represents the target rate ρ. The activation value of the neurons is constrained between the red lines of value $\rho + \beta$ and $\rho - \beta$

$$NRMSE = \frac{\sqrt{\frac{1}{T}\sum_{t=1}^{T}(y[t] - y_{target}[t])^2}}{std(y_{target})} \tag{7}$$

where T is the number of samples, $y[t]$ is the actual value for the t-th sample, $y_{target}[t]s$ is the predicted value for the t-th output of the network and $\sigma(y_{target})$ is the standard deviation of signal y_{target} such that $std(y_{target}) = \sqrt{\frac{1}{T}\sum_{t=1}^{T}(y_{target}[t] - \langle y_{target} \rangle_T)^2}$

III. Analysis of HADSP

A. Interpretation of the rule in a rate base context

Explaining the effect of the previous algorithm in a spiking context [10] was challenging due to the inherently discrete nature of spikes. In the following section, we demonstrate that our rate-based rule is much more interpretable.

For each neuron i, the associated growth indicator Δz_i is linked to its current firing rate x_i.

$$\Delta z_i = \frac{1}{\beta}(\rho - x_i) \tag{8}$$

In order to reach equilibrium, it is necessary for the growth indicator to satisfy the condition $\Delta z_i \in]-1, 1[$, which translates mathematically to the following constraint for equilibrium.

$$x_i \in]\rho - \beta, \rho + \beta[\tag{9}$$

Therefore, the equilibrium condition of the algorithm can be directly related to its hyperparameters β and ρ. This is illustrated in Fig. 2 where the HADSP influence causes the activity of neurons to fall within the specific range that we mentioned.

This interpretation can be extrapolated to spiking neural networks, thus enhancing the comprehension of this set of plasticity in this other context. For our context, we will see in the following question that the behavior of the algorithm is much clearer given this interpretation.

B. Different regimes

As previously mentioned, the hyperparameters of this algorithm include the target rate, ρ, and the growth parameter, β.

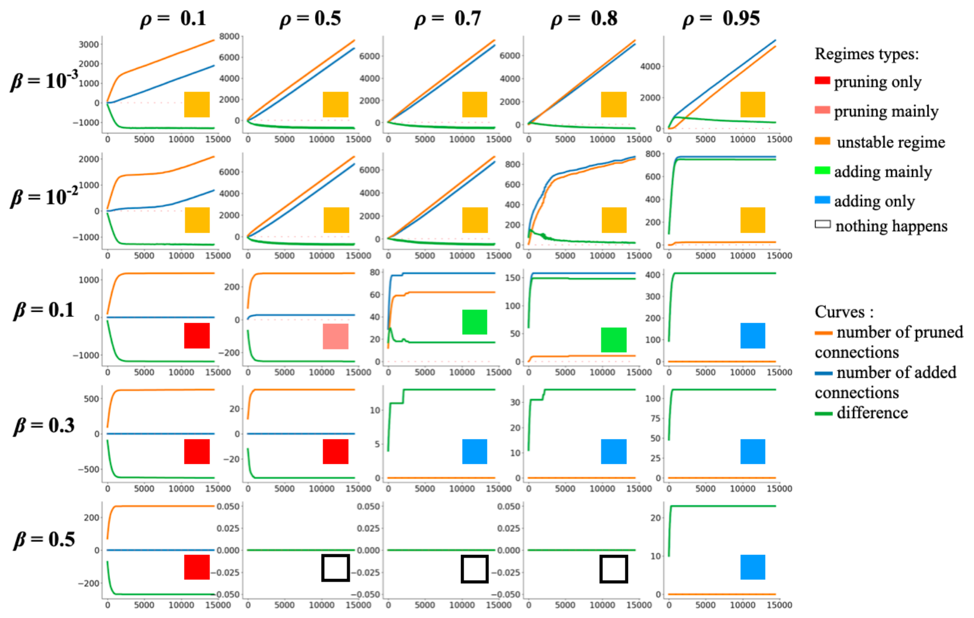

Fig. 3. Different regimes are observed when changing the values of the target rate ρ, and the growth parameter β and the corresponding regime colors. Simulation was done over 15 000 steps for each graph. The y-axis shows the cumulative number of pruned or added connections and the difference between them

One method for gaining insight into the behavior of an algorithm is through the systematic manipulation of its hyperparameters. By varying the values of these parameters and subsequently running the algorithm, distinct regimes within the system can be identified. This approach not only allows for a comprehensive examination of the algorithm's behavior under various conditions, but also facilitates a more thorough understanding of its potential applications.

To probe the behavior we start from a randomly-initialized connection matrix $\mathbf{W}$ sampled from a uniform distribution between $[0, 1]$, bias $\mathbf{b}$ is taken from a normal distribution of mean 0.1 and standard deviation 0.1 and $\mathbf{W_{in}}$ from a normal distribution of mean 1 and standard deviation 0.5. We feed the network with an input of size $1 \times n$ randomly sampled over a sequence length of T from a Poisson distribution that scales in the [0, 1] range. Input scaling is set to 0.1 to ensure that the magnitude of the inputs is low enough to let the neurons combine different entries to reach their target rate ρ. Throughout the experiment, these parameters remain constant and we repeat it for combinations of $\beta \in [0.001, 0.01, 0.1, 0.3, 0.5]$ and $\rho \in [0.1, 0.5, 0.7, 0.8, 0.95]$. We then examine the connectivity changes of the network by tracking the number of connections that are created and pruned at each iteration of the algorithm. The recordings of these values are presented in Fig. 3. The maximum number of partners γ has no influence on the cumulative number of pruned or added connections because if we indeed limit the number of partners, in our model neurons can have several connections which means that γ does not limit the number of connections but rather their distribution. For this experiment we set $\gamma = 12$.

The graphical representation in Fig.3. allows for the visualization of the algorithm's dynamic behavior and also facilitates a more intuitive understanding of the underlying processes.

We have identified five distinct regimes that emerge from the HADSP mechanism, four of which are stable and one is unstable. On some values, we observe that nothing happens which is expected behavior as the target range is too broad and every node activation falls within the range.

The stable regimes can be classified into two categories: pruning plateaus and adding plateaus.

The pruning plateaus correspond to situations where the

TABLE I
SPECTRAL RADIUS OBTAINED AFTER RUNNING THE HADSP ALGORITHM FOR 15 000 STEPS, STARTING FROM A RANDOMLY DRAWN CONNECTION MATRIX $\mathbf{W}$, FOR EVERY COMBINATION OF $\beta \in [0.001, 0.01, 0.1, 0.3, 0.5]$ AND $\rho \in [0.1, 0.5, 0.7, 0.8, 0.95]$. VALUES WERE AVERAGED OVER 10 TRIALS.

		ρ				
		0.1	0.5	0.7	0.8	0.95
β	0.001	0.77 ± 0.04	0.85 ± 0.05	1.07 ± 0.04	1.23 ± 0.02	1.79 ± 0.006
	0.01	0.90 ± 0.05	0.89 ± 0.03	1.10 ± 0.03	1.22 ± 0.02	1.8 ± 0.002
	0.1	2.61 ± 4	0.87 ± 0.004	1.07 ± 0.004	1.20 ± 0.004	1.46 ± 0.002
	0.3	0.62 ± 0.005	0.98 ± 0.001	1.02 ± 0.001	1.05 ± 0.003	1.15 ± 0.004
	0.5	0.83 ± 0.002	1	1	1.03 ± 0.0002	1.03 ± 0.002

initial activity of the network is stronger than the target activity, and the network prunes connections in order to lower its activity. In cases where the input magnitude is particularly strong compared to the target rate, the network may prune all connections until none remain.

On the other hand, the normal pruning process may result in the pruning of too many connections, requiring the creation of new connections to maintain equilibrium. The adding plateaus correspond to situations where the initial activity of the network is weaker than the target activity, and the HADSP mechanism is employed to add connections in order to increase its activity. In some cases, the addition of connections may need to be mitigated through pruning when the overall increase becomes too high.

However, in some instances, the equilibrium of activity cannot be reached due to the narrow range of the equilibrium area, causing the network to continuously oscillate between over and under the threshold of a suitable activity range through quick pruning and adding of connections, leading to an unstable regime.

Generally, if the algorithm fails to converge, the network undergoes constant reorganization, leading to a changing dynamic over time. As a result, for a given input time series, the network will not form a stable representation of the inputs, making it impossible to train using the Ridge regression method. The case β = 0.1, ρ = 0.1 in Table I offers a good example of the unstable representation that can occur from an unstable regime. For those values of β and ρ the spectral radius varies between 0.25 and 12 over our 10 trials.

For values of β that are too low, the target range of activity is too narrow for the algorithm to converge. When β value is too high, the target band is not restrictive enough and the algorithm shows almost no connection creation.

A good way to set the β and ρ is to choose them in order to get a certain spectral radius. In table I we show the spectral radius we achieved for our set of parameters after running the HADSP algorithm for 15 000 steps. As a general guideline for the rest of the experiments, we always set β value to 0.1, which showed the most interesting behavior, and ρ value will be at least 0.7 to ensure the echo state property.

C. *Information processing capacity of HADSP*

In accordance with theory, we used an input sequence sampled from a uniform random distribution in the range [-1,1]. To allow for accurate measurement, we used 10^6 time steps and measured IPC using code provided by [12], details of this implementation can be found in the related paper.

A comparative analysis was conducted to evaluate the performance of networks generated by the HADSP algorithm, starting from an empty connectivity matrix $\mathbf{W_{HADSP}}(t = 0) = \mathbf{0_{n\times n}}$ with parameters $\beta = 0.1$ and $\rho = 0.7$. We examined the impact of the maximum number of partners γ a neuron can have on the IPC for $\gamma \in [2, 8, 12, 20]$. Although γ does not change the dynamics of the scheme discussed earlier, this parameter directly influences the sparsity and weight distribution of the resulting network.

We generate connections by using a pretraining input sampled from the same uniform random distribution in the range [-1,1]. We freeze the weights and we use IPC to assess the computational properties of the resulting ESNs.

We compare those networks with a last network generated using a random distribution as the connectivity matrix. We set the spectral radius and connectivity of this matrix to the same value as the generated HADSP matrix with $\gamma = 12$ to ensure they are comparable. The results presented in Fig. 4 show that that even though the two networks have similar linear processing capacities, their nonlinear processing occurs for lower degrees (less than 4) for the network generated with HADSP. Therefore HADSP creates networks whose computational power is less nonlinear than a randomly drawn network. Our analysis also reveals that as γ increases, the information capacity becomes concentrated in the higher degrees (greater than 4). The relative proportion of computation dedicated to each degree remains largely consistent, suggesting that the effect on computation should not be significant. For the following experiments, we set this value to 12.

An analysis of the linear part of the IPC of the two neural networks, showed in Fig.5 revealed that they should show similar performances on memory-related tasks. These findings are consistent with the memory capacity curves previously reported in the literature. A curve similar to Fig. 5 can be seen in [8]. It was also observed that the network generated using the HADSP method exhibited a higher percentage of computation for lower degree operations. Therefore, it can be inferred that in scenarios where the task at hand does not require a significant degree of non-linearity, the HADSP-generated network may exhibit superior performances.

IV. Analysis of Hebbian mechanism

A. *Hebbian like plasticity*

We now discuss a 3 phased scenario to illustrate how Hebbian plasticity can emerge from HADSP rule.

Step 1 : We start from a blank network $\mathbf{W} = \mathbf{0_{n\times n}}$ of $n = 100$ excitatory neurons, bias $\mathbf{b}$ sampled from a normal distribution of mean 0.1 and standard deviation 0.1, target rate

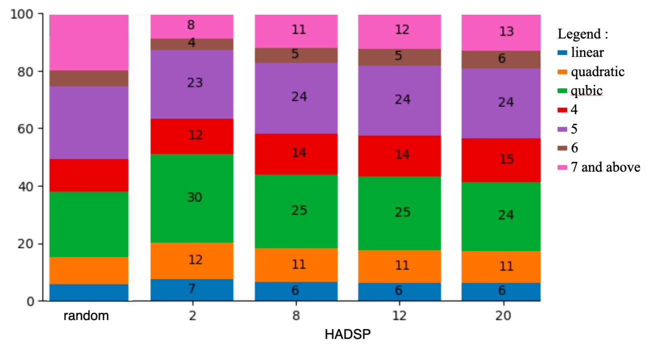

Fig. 4. Measured IPC in terms of non-linearity for a randomly generated network generated network and an HADSP generated networks for different values of the maximum number of partners γ. Values were averaged over 10 independent generations

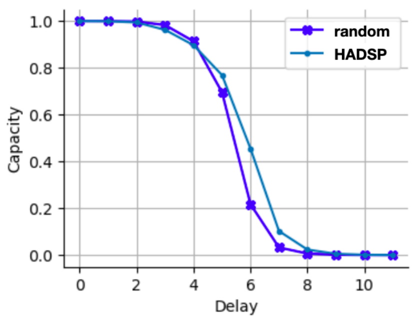

Fig. 5. Memory curves for linear component only, averaged were taken over 10 independent trials.

ρ is set to 0.7 and a growth rate β to 0.1. The input value is generated as white noise sampled from a Poisson distribution. This leads the neurons to remodel their connections following the HADSP algorithm until an equilibrium is reached.

Step 2 : From this equilibrium situation, we clamp a slightly enhanced input (relatively to the remainder of the network) on a portion of the network (neurons 51 to 60). This forces the over-stimulated neurons to prune their incoming connections.

Step 3 : After equilibrium is reached again, the enhanced input is removed, and the network is driven solely by input noise again. The activity-dependent structural plasticity algorithm then leads the previously stimulated neurons to establish connections with each other.

In conclusion, this scenario, although exaggerated here for the sake of demonstration, highlights that neurons regulated by HADPS can developed intricate connections based on their previous correlated over-stimulation. This outcome shows that our algorithm, although purely homeostatic, display a form of Hebbian learning.

We hypothesize that this Hebbian mechanism can improve network performance. When presented with inputs that are similar, the network can combine those inputs to create new and more useful dynamics.

B. Prediction of chaotic time series

The inputs we have used so far did not contain any inherent information. To further validate our hypothesis that the HADSP algorithm can generate meaningful combinations of otherwise redundant inputs, we turn to the popular benchmark of Mackey Glass equations. The purpose is not to focus on the performance on Mackey glass but rather to demonstrate that the HADSP algorithm can exploit correlation in input to improve performance. Therefore we didn't systematically optimise the hyperparameters of our ESNs.

As previously explained, the equations describing the network are 1 and 2. We use a ”warm up” sequence prior to training the reservoir. The training of the reservoir is then done to forecast the subsequent time-step of the time-series given solely the current time-step. The training procedure utilizes ridge regression with a commonly efficient parameter of $r = 10^{-7}$. After training, the last state of the reservoir during the training sequence is used the initial state for the prediction sequence.

TABLE II
NRMSE AVERAGED OVER 10 TRIALS, FOR MG-17 AND MG-30 BENCHMARKS FOR A NETWORK GENERATED RANDOMLY AND THROUGH HADSP. THE NRMSE WAS CALCULATED USING 300 STEPS FOR THE MG-17 AND 100 STEPS FOR THE MG-30

Model	τ	HADSP	random
MG17	1	0.021	0.0075
	30	0.57	0.52
	100	0.73	0.67
MG30	1	0.0060	0.0043
	10	0.53	0.50
	30	0.41	0.40

We let the network evolve for the pre-training time period in an unsupervised manner utilizing the HADSP algorithm with $\rho = 0.8$ and $\beta = 0.1$. As previously explained those values are chosen to obtain a spectral radius that will ensure that the networks produce a stable representation of their input, meaning that the regime of connection formation converges and that the dynamics of the network are healthy. For this experiment, in order to capture the correlation in the time series we choose a smaller increment of value 20 for our rule, which means that the computation of $\mathbf{Deltaz}$ and the following addition and pruning happen every 20 steps. Once the plastic unsupervised learning process is completed, the output weight matrix W_{out} is calculated in a supervised manner utilizing the generated ESN to reproduce the signal τ steps ahead.

We compare this first network with matrix $\mathbf{W_{HADSP}}$ against a additional networks $\mathbf{W_{random}}$ of equal size, $n = 260$, with same connectivity but whose with weights sampled from a uniform distribution between 0 and 1, we scale the $\mathbf{W_{random}}$ matrix so that its spectral radius is the same than $\mathbf{W_{HADSP}}$. For the two networks the $\mathbf{W_{in}}$ input matrix is taken from a normalize distribution of mean 1 and standard deviation 0.5 with connectivity of 0.5. For the two networks the $\mathbf{W_{in}}$ input matrix is taken from a normalize distribution of mean 1 and standard deviation 0.5.

The results in Table II show that for the version of Mackey Glass the HADSP algorithm never shows better performances compared to the randomly instanciated network.

We use a modified approach for this benchmark where the input is transformed into 130 separate inputs by using Butterworth bandpass filters of a selected set of frequencies. The idea is that from inputs that are similar, and thus do not add to the overall richness of the reservoir, the HADSP will be able to combine them into a more useful dynamic. We have used the specific frequencies reported in [14] as band frequencies for our model. A bandpass filter need a low and high frequency to be defined. The lowest frequencies are:

$f_{low} = [0, 0.04, 0.015 : 0.02 : 0.075, 0 : 0.001 : 0.0995, 0.05 : 0.005 : 0.15, 0.15 : 0.1 : 0.55]$ π rad/sample

Where $a : b : c$ means that the frequencies are ranged from a to c with a step of size b.

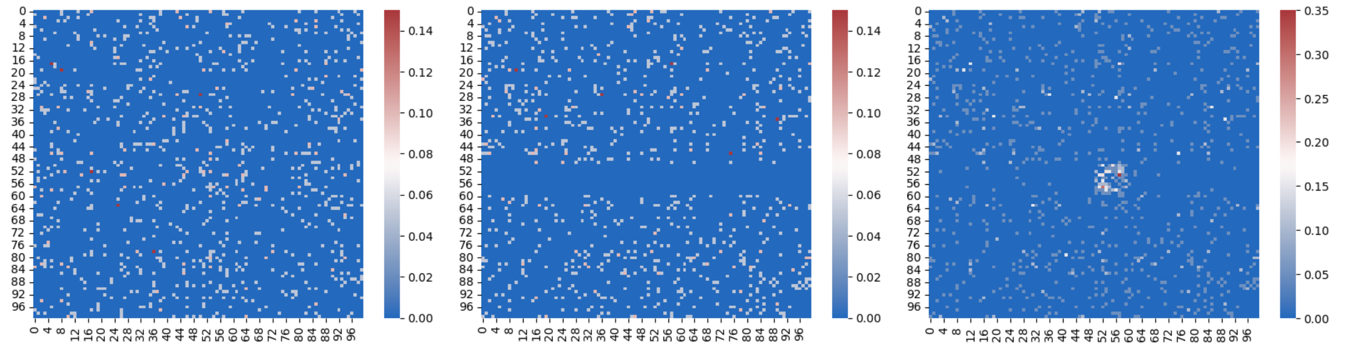
Fig. 6. Evolution of the connectivity matrices during the experiment: After starting from a blank connection matrix, the neurons create connections between themselves through a homeostatic mechanism. Larger inputs on neurons 51 to 60 force these neurons to prune their weights in order to achieve homeostasis. Once the larger input is removed, the other neurons have already reached homeostasis, and neurons 51 to 60 mostly create connections within the depleted area.

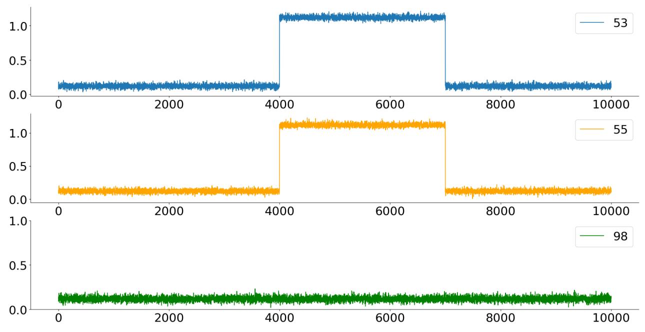

Fig. 7. Typical input curves for our experiment, here nodes 53 and 55 simultaneously received a higher input than node 98 which will eventually result in 53 and 55 being preferentially connected.

And the highest frequencies are given by the following relation:

$$f_{high} = \begin{cases} f_{low} + 0.01 & \text{if } f_{low} < 0.05 \\ f_{low} + 0.1 & \text{if } f_{low} \geq 0.05 \end{cases} \tag{10}$$

We obtain 130 different band-filtered inputs which examples can be seen in Fig. 10. The size of the network was chosen accordingly to $n = 260$ so that each incoming input signal is assigned to only two nodes in the reservoir. The connectivity matrix $\mathbf{W_{in}}$ input matrix is taken from a normal distribution of mean 1 and standard deviation 0.5. Fig.9 summarizes the structure of the experiment.

We generate a connection matrix through the HADSP mechanism $\mathbf{W_{HADSP+band}}$ of size $n = 260$ by the same methods as the previous one with $\rho = 0.8$, $\beta = 0.1$, $\gamma = 12$, and the increment of the algorithm set to 20 steps. Once again, when the plastic unsupervised learning process has been completed, the output weight matrix W_{out} is calculated in a supervised manner utilizing the generated ESN and the 130 filtered inputs to reproduce the original Mackey glass signal τ steps ahead.

Again we compare this network to a second matrix $\mathbf{W_{random+band}}$ that we generate from a uniform distribution between 0 and 1 with the same connectivity, we scale the $\mathbf{W_{random+band}}$ matrix so that its spectral radius is the same as $\mathbf{W_{HADSP+band}}$. used the same inputs as the first network, and the $\mathbf{W_{in}}$ input matrix is taken from a normal distribution of mean 1 and standard deviation 0.5 with a connectivity of 1.

In this second study, the HADSP algorithm consistently demonstrated superior performance to the other randomly generated network for the MG-17 and MG-30 datasets, across various steps ahead value τ. For the MG-30 dataset, the results were less pronounced but the HADSP algorithm still achieved favorable results even for larger time steps.

Those improvements were not due to hyperparameters tuning and show that HADSP can improve performance on specific types of data. Our interpretation is that the network with HADSP was able to capture the correlation in the band-

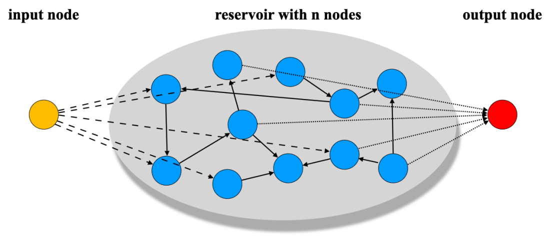

Fig. 8. Setup for our experiments for the classic architecture

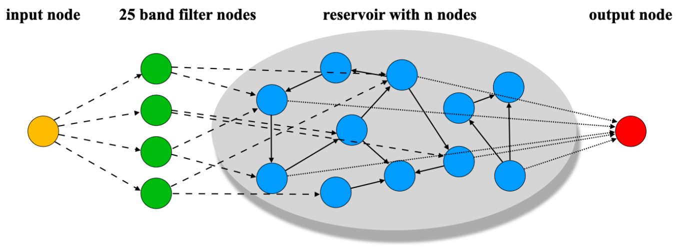

Fig. 9. Setup for our experiments when we use band-filtered inputs for the reservoir

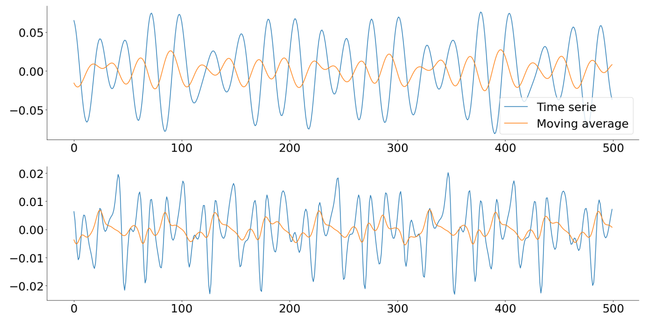

Fig. 10. Typical time series and moving average obtained by applying a band filter on the inputs

TABLE III
NRMSE AVERAGED OVER 10 TRIALS, FOR MG-17 AND MG-30 BENCHMARKS USING BAND-PASS FILTERS FOR A NETWORK GENERATED RANDOMLY AND ANOTHER+ THROUGH HADSP. THE NRMSE WAS CALCULATED USING 300 STEPS FOR THE MG-17 AND 100 STEPS FOR THE MG-30

Model	τ	HADSP+band	random + band
MG17	1	0.0022	0.0040
	30	0.096	0.10
	100	0.44	0.47
MG30	1	0.035	0.038
	10	0.15	0.17
	30	0.24	0.24

filtered data and use it to improve performance.

V. Conclusion

We proposed a novel rate-based variant of homeostatic activity-dependent structural plasticity (HADSP), drawing inspiration from previous literature in computational neuroscience and spiking neural networks. This rate-based model can reproduce the key behavior of its spiking equivalent and is much more interpretable. We have provided a clear explanation of the interpretability of our rate-based rule by explicitly describing how the hyperparameters constrain the behavior of the network.

We postulated that this rate-based HADSP algorithm can enhance the dynamics of a reservoir computing network by combining redundant inputs into unique and useful ones. To test this hypothesis, we present a thorough examination of the computational properties of networks generated entirely or partially with HADSP. This approach allowed us to prove that the networks created through HADSP show indeed different computational behavior than the ones instantiated through random uniform distributions, in particular networks generated with HADSP exhibit a lower degree of nonlinearity compared to randomly generated networks. We further validated our hypothesis by applying our network to the Mackey Glass benchmark. Our results show that when two HADSP networks are presented with either the Mackey Glass time series or an equivalent representation of multiple band-filtered Mackey Glass inputs, the latter one exhibits better performance. We conclude that HADSP effectively leverages the inter-relationships of network inputs to achieve richer dynamics, which is a result of the network's ability to synthesize inputs that reflect similar dynamics.

Our findings demonstrate the potential for a novel rate-based variant of homeostatic activity-dependent structural plasticity to effectively leverage the inter-relationship of network inputs.

For future work, we plan to expand on this promising direction and systematically evaluate the performance of the algorithm on a diverse set of tasks and benchmarks, including other plasticity-based network constructions. This will allow us to gain a deeper understanding of the effectiveness and limitations of the algorithm. Furthermore, we plan to explore the conditions under which the algorithm can improve performance and investigate the underlying mechanisms behind this improvement, in particular the complementarity between Hebbian learning and homeostasis mechanisms. Our goal is to establish a comprehensive understanding of the rate-based HADSP algorithm and its potential applications in the field of reservoir computing.

References

[1] Jianhua Cang and David A. Feldheim. Developmental mechanisms of topographic map formation and alignment. *Annual Review of Neuroscience*, 36(1):51–77, jul 2013.

[2] Jill Sakai. How synaptic pruning shapes neural wiring during development and, possibly, in disease. *Proceedings of the National Academy of Sciences*, 117(28):16096–16099, jun 2020.

[3] Paul Miller. The role of structural plasticity in producing nonrandom neural connectivity. In *The Rewiring Brain*, pages 221–245. Elsevier, 2017.

[4] Markus Butz, Florentin Wörgötter, and Arjen van Ooyen. Activity-dependent structural plasticity. *Brain Research Reviews*, 60(2):287–305, may 2009.

[5] Mantas Lukoševičius. A practical guide to applying echo state networks. In *Lecture Notes in Computer Science*, pages 659–686. Springer Berlin Heidelberg, 2012.

[6] Gregor M. Hoerzer, Robert Legenstein, and Wolfgang Maass. Emergence of complex computational structures from chaotic neural networks through reward-modulated hebbian learning. *Cerebral Cortex*, 24(3):677–690, nov 2012.

[7] Andreea Lazar. SORN: a self-organizing recurrent neural network. *Frontiers in Computational Neuroscience*, 3, 2009.

[8] Guillermo B. Morales, Claudio R. Mirasso, and Miguel C. Soriano. Unveiling the role of plasticity rules in reservoir computing. *Neurocomputing*, may 2021.

[9] Michael Fauth and Christian Tetzlaff. Opposing effects of neuronal activity on structural plasticity. *Frontiers in Neuroanatomy*, 10, jun 2016.

[10] Júlia V. Gallinaro and Stefan Rotter. Associative properties of structural plasticity based on firing rate homeostasis in recurrent neuronal networks. *Scientific Reports*, 8(1), feb 2018.

[11] Markus Butz and Arjen van Ooyen. A simple rule for dendritic spine and axonal bouton formation can account for cortical reorganization after focal retinal lesions. *PLoS Computational Biology*, 9(10):e1003259, oct 2013.

[12] Joni Dambre, David Verstraeten, Benjamin Schrauwen, and Serge Massar. Information processing capacity of dynamical systems. *Scientific Reports*, 2(1), jul 2012.

[13] H. Jaeger. Short term memory in echo state networks. 2001.

[14] Francis Wyffels, Benjamin Schrauwen, David Verstraeten, and Dirk Stroobandt. Band-pass reservoir computing. In *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*. IEEE, jun 2008.

[15] Michael C. Mackey and Leon Glass. Oscillation and chaos in physiological control systems. *Science*, 197(4300):287–289, jul 1977.

[16] Chenxi Sun, Moxian Song, Shenda Hong, and Hongyan Li. A review of designs and applications of echo state networks. December 2020.