This is a repository copy of *Modelling and evaluating restricted ESNs on single- and multi-timescale problems*.

White Rose Research Online URL for this paper:
https://eprints.whiterose.ac.uk/218912/

Version: Published Version

Article:
Wringe, Chester, Stepney, Susan orcid.org/0000-0003-3146-5401 and Trefzer, Martin Albrecht orcid.org/0000-0002-6196-6832 (2024) Modelling and evaluating restricted ESNs on single- and multi-timescale problems. Natural Computing. ISSN 1567-7818

https://doi.org/10.1007/s11047-024-10004-7

Reuse
This article is distributed under the terms of the Creative Commons Attribution (CC BY) licence. This licence allows you to distribute, remix, tweak, and build upon the work, even commercially, as long as you credit the authors for the original work. More information and the full terms of the licence here: https://creativecommons.org/licenses/

Takedown
If you consider content in White Rose Research Online to be in breach of UK law, please notify us by emailing eprints@whiterose.ac.uk including the URL of the record and the reason for the withdrawal request.

eprints@whiterose.ac.uk
https://eprints.whiterose.ac.uk/

Natural Computing
https://doi.org/10.1007/s11047-024-10004-7

Modelling and evaluating restricted ESNs on single- and multi-timescale problems

Chester Wringe[1] · Susan Stepney[1] · Martin A. Trefzer[2]

Accepted: 10 October 2024
© The Author(s) 2024

Abstract
Reservoir Computing is a computing model ideal for performing computation on varied physical substrates. However, these physical reservoirs can be difficult to scale up. We propose joining various reservoirs together as an approach to solving this problem, simulating physical reservoirs with Echo State Networks (ESNs). We investigate various methods of combining ESNs to form larger reservoirs, including a method that we dub *Restricted ESNs*. We provide a notation for describing Restricted ESNs, and use it to benchmark a standard ESN against restricted ones. We investigate two methods to keep the weight matrix density consistent when comparing a Restricted ESN to a standard one, which we call *overall consistency* and *patch consistency*. We benchmark restricted ESNs on NARMA10 and the sunspot prediction benchmark, and find that restricted ESNs perform similarly to standard ones. We present some application scenarios in which restricted ESNs may offer advantages over standard ESNs. We then test restricted ESNs on a version of the multi-timescale Multiple Superimposed Sines tasks, in order to establish a baseline performance that can be improved upon in further work. We conclude that we can scale up reservoir performance by linking small homogeneous subreservoirs together without significant loss in performance over a single large reservoir, justifying future work on using heterogeneous subreservoirs for greater flexibility.

Keywords Reservoir computing · Hierarchical ESNs · Reservoir of reservoirs · Multiple superimposed oscillators

1 Introduction

Artificial Neural Networks (ANNs) are an unconventional computational model inspired by the brain. ANNs have non-linear summing nodes connected by weighted edges, where the edge weights are trained to give the desired outputs. While training methods such as backpropagation are used in feed-forward Neural Networks, they are costly to use in recurrent NNs (RNNs).

Reservoir Computing in general (Jaeger 2001; Maass et al. 2002; Jaeger et al. 2007), and the Echo State Network (ESN) random RNN model in particular, provides a solution to the RNN training problem: instead of training the recurrent, inner weights, these are randomly initialised, and only the weights of the edges to the output nodes are trained. This provides an efficient training method, and also allows the inner network (or "reservoir") to be treated as a black box. One may use any material or substrate as an *in materio* reservoir.

In materio Computing (Harding and Miller 2004) is the term for computation performed using a direct mapping of a computational model to a physical material[1]. Early instantiations of *in materio* computing involved evolving physical configurations of a given substrate to perform a given task, such as signal classification (Thompson and Layzell 1999).

✉ Chester Wringe
chester.wringe@york.ac.uk

✉ Susan Stepney
susan.stepney@york.ac.uk

✉ Martin A. Trefzer
martin.trefzer@york.ac.uk

1 Department of Computer Science, University of York, York, UK

2 School of Physics, Engineering and Technology, University of York, York, UK

[1] See Stepney (2024) for a discussion of the difference between *in materio* and classical computing (section 1.1) and what it means for a physical system to compute (Sect. 2).

Published online: 26 October 2024

Reservoir Computing is a powerful model for in materio computing, as it can be mapped to any physical substrate with sufficiently rich dynamics. One of the earliest works in the field involved using a bucket of water as a reservoir (Fernando and Sojakka 2003).

Physical RC has been reviewed extensively by Tanaka et al. (2019), Zhang and Vargas (2023) review a number of physical RC systems grouped by application. Tutorials on how to implement physical RCs have been written by Stepney (2024) and Cucchi et al. (2022).

Our long term aim is to scale up the capacity of *in materio* reservoirs by combining several reservoirs with differing properties. Combining reservoirs has some potential advantages: it allows us to more fully exploit substrates whose computational capacity does not scale well as the size of the device increases (Dale et al. 2021), and to exploit heterogeneous substrates with different properties for more complex tasks, particularly those with multiple timescales.

Here, we focus on homogeneous reservoirs, to compare performances of multiple connected small reservoirs against a single larger one, and to provide a baseline for future work. We introduce a notation for describing a form of reservoir combination. We perform some experiments using the ESN model, using NARMA-10, sunspots, and MSO benchmarks.

2 Background

2.1 Reservoir computing

Two ways of combining multiple reservoir computers emerge in the literature. The first, which we call *modular* reservoirs (Sect. 2.2), are larger systems that contain within them multiple reservoir computers, each with their own set of inputs and individually trained output weights. The second, which we call *restricted ESNs* (Sect. 2.3) are the subject of our experiments here.

2.2 Modular ESNs

A *modular* ESN typically comprises multiple individual reservoirs, each with its own input layer and trained output weights, connected in a variety of ways.

The Dynamic Feature Discoverer (DFD) (Jaeger 2007) is a modular reservoir based on Deep Belief Networks, with the ESNs being components of a larger system. The ESNs may be replaced by other components, such as Extreme Learning Machines. The ESNs are arranged hierarchically, with each ESN being fed the standard input as well as the outputs of all the ESNs lower in the hierarchy. This hierarchy also allows the DFD to contain separate timescales, such that each ESN in one level of the hierarchy runs more slowly than those in the previous levels.

Modular ESNs are also used in acoustic modelling (Triefenbach et al. 2010, 2013). This model is based on the Hidden Markov Model, with the different ESNs with different timescales arranged linearly and hierarchically, with each reservoir processing dynamics that are slower than the previous ones.

The ConvESN (Ma et al. 2021) is a modular reservoir model based on Convolutional Neural Networks. The reservoirs are arranged in parallel and analyse dynamics at different timescales. The trained outputs of the ESN are then joined together in a convolutional layer.

2.3 Restricted ESNs

A *restricted ESN* has the same overall structure as a single ESN, with one input layer and one output layer. Its internal reservoir (a random RNN in the ESN model) has its overall state partitioned into "subreservoirs" with typical RNN connections within a subreservoir, and *restricted* connections between the subreservoirs. There are several models in the literature that follow this structure.

The dual-reservoir network (DRN) (Ma et al. 2017b) connects two subreservoirs in the network with an "unsupervised encoder", for which the weights are chosen using Principal Component Analysis (PCA). Triefenbach et al. (2013) have a bidirectional dual-reservoir model, which consists of two subreservoirs running in parallel, with one of the subreservoirs receiving the inputs in chronological order, and the other receiving its inputs in reverse chronological order.

The Reservoir of Reservoirs (RoR) (Dale 2018a) is a model with dense connections within each subreservoir, and sparse random connections between subreservoirs. Two models are investigated: RoR, where the inputs are sent to only one subreservoir, and RoR-IA, where the inputs are sent to all of the subreservoirs. The multilayered echo state machine (ML-ESM) (Malik et al. 2017) arranges the subreservoirs sequentially, with each subreservoir fully connected to its neighbouring subreservoirs with fixed weights.

The Reservoir with Random Static Projections (R^2SP) (Butcher et al. 2010) and the ϕESN (Gallicchio and Micheli 2011) are both models that combine ESNs with an Extreme Learning Machine. There are several deep-ESN models (Gallicchio and Micheli 2017; Gallicchio et al. 2017; Ma et al. 2017a; Canaday et al. 2021) based on deep learning networks. In these, the subreservoirs are arranged sequentially, and the inputs are sent only to the first subreservoir. They are compared to the grouped-ESN, where

the subreservoirs are arranged in parallel, and deep-ESN Input-to-All (deep-ESN IA), a deep-ESN with inputs sent to every subreservoir.

Iinuma et al. (2022) use a restricted ESN model called the Assembly ESN (aESN) in order to extract features from multiple inputs. The reBASIC model (Kawai et al. 2023a, b) is another ESN model that follows the restricted ESN architecture.

The Decoupled ESN (DESN) (Xue et al. 2007) is a restricted ESN that tackles multi-timescale tasks by decoupling certain sections of the inner state from each other using a lateral inhibition unit.

The scale-free highly clustered ESN (SHESN) (Deng and Zhang 2007) has each subreservoir connected to every other subreservoir by "backbone nodes", of which there is one in every subreservoir. The hierarchically clustered ESN (HESN) (Jarvis et al. 2010) builds on the SHESN by allowing several backbone nodes per subreservoir, and by making them randomly connected as opposed to fully connected. The HESN and the modular ESN (Rodriguez et al. 2019) are the closest models to the ones we study here.

3 The restricted ESN model

Here we investigate the restricted ESN (rESN) model. This provides a model that should allow for the simulation of *in materio* subreservoirs implemented with different materials, with some physical interconnect between subreservoirs. We introduce a notation that can be used to describe a variety of possible restrictions that may occur in practice, including the models reviewed in Sect. 2.3.

3.1 The standard ESN model

The original ESN model (Jaeger 2001; Jaeger et al. 2007) (Fig. 1) is a Random RNN where only the output weights are trained. A standard ESN can be described by three state vectors and three weight matrices (corresponding to input, internal, and output states and weights), and a set of state update equations.

At time t, the state of the ESN is described by the input vector $\mathbf{u}(t)$, the internal state vector $\mathbf{x}(t)$, and the output vector $\mathbf{v}(t)$. The connections between the nodes represented by the vectors are described by the weight matrices $\mathbf{W}_u$ for the random input weights, $\mathbf{W}$ for the random internal weights, and $\mathbf{W}_v$ for the trained output weights.

We use the update equations for the ESN from Stepney (2021):

$$\begin{aligned} \mathbf{x}(t+1) &= f(\mathbf{W}_u\mathbf{u}(t) + \mathbf{W}\mathbf{x}(t)) \\ \mathbf{v}(t+1) &= \mathbf{W}_v\mathbf{x}(t) \end{aligned} \tag{1}$$

where f is a nonlinear function, typically the hyperbolic tangent tanh(.).

3.2 Restricting the standard model

The rESN is a variant of the standard ESN model that divides the internal reservoir state $\mathbf{x}$ into several smaller subreservoir states. This division may be interpreted as restrictions on the connections between parts of the internal state, and thus on the internal weight matrix $\mathbf{W}$. The state vector $\mathbf{x}$ of a restricted ESN with n subreservoirs is the concatenation of the subreservoir state vectors:

$$\mathbf{x} = \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \dots \\ \mathbf{x}_n \end{pmatrix} \tag{2}$$

where $\mathbf{x}_i$ is the state of the subreservoir i. N_i is the number of nodes in subreservoir i; $N = \sum_{i=1}^{n} N_i$ is the number of nodes in the entire state.

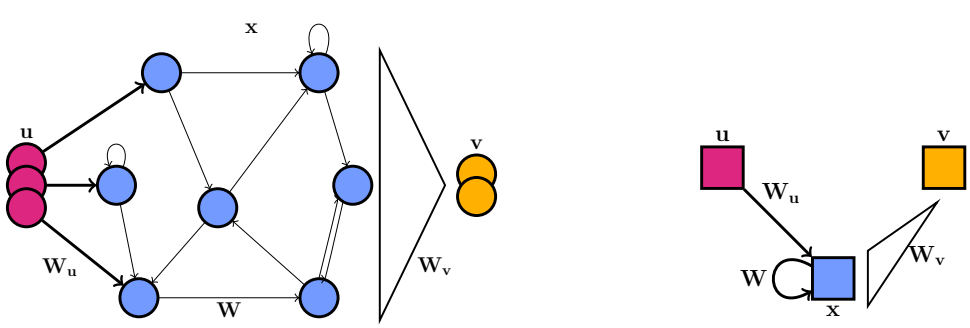

Fig. 1 An example of a standard ESN with $n = 7$ nodes (**a**) and an abstraction of its different elements (**b**). The ESN takes one or more inputs (**u**) which are then sent to the inner state (**x**) through weighted edges ($\mathbf{W_u}$). The weights within the inner state (**W**) are recurrent and randomly initialised. Finally, the output state (**v**) receives the inner state through the trained output layer $\mathbf{W_v}$

The weight matrix is the concatenation of internal subreservoir weight matrices, and weight matrices describing the connections between subreservoirs:

$$\mathbf{W} = \begin{pmatrix} \mathbf{W}_1 & \mathbf{B}_{12} & \dots & \mathbf{B}_{1n} \\ \mathbf{B}_{21} & \mathbf{W}_2 & \dots & \mathbf{B}_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{B}_{n1} & \mathbf{B}_{n2} & \dots & \mathbf{W}_n \end{pmatrix} \tag{3}$$

where $\mathbf{W}_i$ is the weight matrix that represents the connections within subreservoir i, and $\mathbf{B}_{ij}$ represents the connections from subreservoir i to subreservoir j; $\mathbf{B}_{ij}$ is square if subreservoirs i and j have the same size. The output and input weight matrices are unchanged.

These elements are illustrated in Fig. 2, using the notation introduced in Fig. 1b. Equation 1 still defines the transfer from the overall state at time t to time $t+1$.

In general, the submatrices may each have their own, independent properties such as connection density D, the proportion of non-zero weight values. Here we consider uniform subreservoirs (all the $\mathbf{W}_i$ have the same average densities D_W) and uniform connectivities (all the $\mathbf{B}_{ij}$ have the same average densities D_B).

4 Density experiments

We are developing this model in order to provide a means to join *in materio* reservoirs with different properties and different timescales. Before investigating such heterogeneous systems, however, we need to investigate homogeneous restricted reservoirs, to determine the effect of restriction alone. Does an rESN (with its N nodes partitioned into loosely connected subreservoirs) perform significantly differently from a standard ESN of the same dimension (a single reservoir of N nodes)?

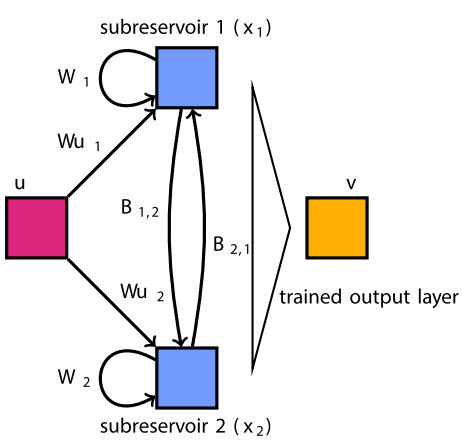

Fig. 2 Elements of an rESN with 2 subreservoirs, 1 and 2, showing the partitioned state and components of the internal weight matrix

In order to test this question, we must determine what constitutes a fair comparison between an rESN and a standard ESN with the same number of nodes. We perform a comparison of the two models over a range of different sizes, on two common benchmark tasks.[2]

4.1 Experimental setup

We wish to discover whether any difference in performance found is due merely to the architecture, or to some other parameter affected by the restriction.

We further wish to ensure that the standard ESN and rESN can each exhibit their best performance on the given task; however, what this entails is not obvious. In the case of the standard ESN, we may perform a simple search to find some "optimal" weight matrix density for the task. Given this optimised density, we investigate two options for the rESN, which we call *patch consistency* and *overall consistency*.

4.1.1 Patch-consistent density

For this approach, we use a physical analogy to describe the structure of the rESN. If we see this restriction as directly combining multiple physical (material) reservoirs, then restricting a standard reservoir is analogous to having multiple small pieces of a material, and joining these together, in order to emulate a larger reservoir. As such, we should keep the density within the subreservoirs consistent with the overall density of the standard ESN, with sparser connections between subreservoirs. This is illustrated in Fig. 3b.

4.1.2 Overall-consistent density

For this approach, we use a neuronal analogy to describe the structure of the rESN, with an underlying neural network architecture being "rewired". Unlike the patch-consistent approach, this does not lead to a lower overall density of the restricted ESN. Having found the optimal connection density for a standard ESN, we redistribute the edges, moving some from the $\mathbf{B}$ weight matrices to the $\mathbf{W}$ weight matrices, so that there are more connections within subreservoirs than outside them, while maintaining a constant number of edges (Fig. 3c). Thus, the overall density of the rESN remains the same as the density of the standard ESN, while ensuring the constraints on topology that makes it an rESN.

[2] Preliminary results for these two benchmarks are reported in Wringe et al. (2023). Here we extend those results to include larger reservoirs (up to 512 nodes), more subreservoirs (up to 8 subreservoirs), and a further benchmark (MSO).

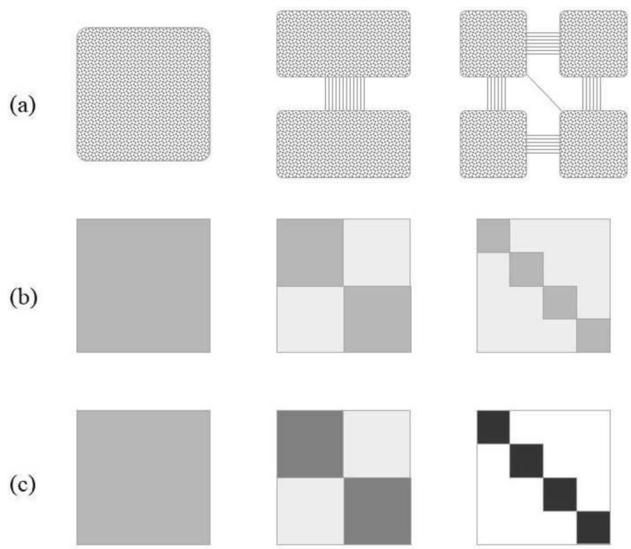

Fig. 3 Illustration of the two density options: **a** a hypothetical physical reservoir, as one large piece of material, or divided into two or four smaller loosely connected pieces; **b** corresponding model patch-consistency weight matrix for a material with a weight matrix density D: each subreservoir has the same material density $D_W = D$, and there is a lower connection density $D_B < D$ between subreservoirs; **c** overall-consistency weight matrix: the total number of connections is constant, so the subreservoirs get increasingly higher densities, and the subreservoir connections get increasingly lower densities

4.2 Benchmarks

In order to determine optimal densities and evaluate the reservoir models, we use two benchmarks, NARMA-10 (an open system, or driven system, task) and Sunspots (a closed system task). All training is performed using ridge–regression.

4.2.1 NRMSE

The results are reported as the Normalised Root Mean Square Error (Lukoševičius 2012) evaluated over 50 runs.

$$NRMSE(\hat{\mathbf{v}}, \mathbf{v}) = \sqrt{\frac{\langle(\hat{\mathbf{v}} - \mathbf{v})^2\rangle}{\langle(\hat{\mathbf{v}} - \langle\hat{\mathbf{v}}\rangle)^2\rangle}} \tag{4}$$

where $\hat{\mathbf{v}}$ is the desired output; $\mathbf{v}$ is the observed output; $\langle x\rangle$ is the mean $\frac{1}{N}\sum_{i=1}^{N} x_i$.

4.2.2 NARMA10

The Normalised Auto-Regressive Moving Average (NARMA) tasks are a family of benchmark tasks (Atiya and Parlos 2000) frequently used as a reservoir computing benchmark. Here, we use NARMA10, the system with a memory of 10 timesteps:

$$x(t+1) = 0.3x(t) + 0.05x(t)\sum_{i=0}^{9} x(t-i) + 1.5u(t-9) + 0.1 \tag{5}$$

The input at time t, $u(t)$, is uniformly sampled between 0 and 0.5. We use a training length of 3000 data points and washout and testing lengths of 1000 data points each.

4.2.3 Sunspots

The Sunspots benchmark is a dynamical systems benchmark task that involves predicting the next output of the dataset based on the previous outputs. This task has a long history of being used in machine learning generally (Yule 1927), as well as reservoir computing specifically (Schwenker and Labib 2009; Rodan and Tino 2011; Stepney 2021).

We use the monthly readings from the Zurich dataset,[3] from January 1749 to December 1983. As the existing data limits our input lengths, the training length for this experiment is 1500 data points, with a washout length of 500 data points, and a testing length of 820 data points.

4.3 Optimal density

To find the optimal density D_O for a standard ESN on a given benchmark, we use a two-level grid search (Algorithm 1).

Algorithm 1 Optimal density for standard ESN

```
procedure GRIDSEARCH(start, end, step, N)
                                ▷ search over densities d
    for d in (start, end, step) do
        testsum := 0
        for run in range N do
            create ESN with density d
            v := test ESN on benchmark
            testsum += NRMSE(vhat, v)
        means[d] := testsum / N
    return means

coarse := GRIDSEARCH(0, 1, 0.1, 50)
d1 := argmin(coarse)
d2 := density of the smaller of d1's neighbours
fine := GRIDSEARCH(d1, d2, 0.01, 50)
return min(fine)
```

[3] https://machinelearningmastery.com/time-series-datasets-for-machine-learning/

For the patch-consistent rESN, the density within each subreservoir, D_W, is set equal to D_O, while a further two-level grid search is used to find the optimal density between subreservoirs, D_B. In this case, the algorithm is modified to use a step of 0.025 for the first level, and 0.0025 for the second. We also change the *start* and *end* values in the GRIDSEARCH procedure. We set the starting density in our search to $(n/N)^2$, where n is the number of subreservoirs, and N the number of nodes in the full reservoir. This creates a lower bound for D_B, in order to ensure that every connection weight matrix $\mathbf{B}_{ij}$ has at least one entry on average. Our end value for the search is $D_W/4$. This is to ensure that D_B is materially different from D_W, as if no such constraint is set, then the optimal value for D_B is simply D_W.

For the overall-consistent rESN, we introduce a parameter $f = D_W/D_B > 1$, specifying how much higher we wish the internal density in the subreservoirs to be, compared to the connections between them. We then derive D_B and D_W in terms of this f, the overall optimal density D_O, and the number of subreservoirs n (see Appendix A). There is an upper bound on the value of f: too high and it is impossible to achieve the desired weight ratio for a given number of connections (see Appendix B). Given this upper bound for possible f values, we use a similar two-level grid search[4] to find the best f value for a reservoir of size N, density D_O, for the given benchmark.

Having found the optimal densities and distributions, we then evaluate the standard and restricted reservoirs against the task over 50 runs. The experiments are performed for ESNs of size $N \in [64, 128, 256, 512]$, and with 2, 4, and 8 equal-sized subreservoir restricted ESNs.

The densities used for each size for each task are given in Tables 1 and 2.

Table 1 Densities used in the NARMA experiments, for 2, 4, and 8 subreservoirs

N	D_W	D_B		
		2	4	8
64	0.1	0.0010	0.0039	0.015
128	0.1	0.0002	0.0010	0.0039
256	0.1	1.5×10^{-5}	0.0002	0.0010
512	0.1	1.5×10^{-5}	6.1×10^{-5}	0.0002

N	D_O	f		
		2	4	8
64	0.1	114.57	10.84	6.18
128	0.1	491.32	147.02	138.76
256	0.1	459.47	524.01	325.48
512	0.1	1311.52	3603.28	981.64

(top) patch-consistent; (bottom) overall-consistent

Table 2 Densities used in the Sunspots experiments, for 2, 4, and 8 subreservoirs (top) patch-consistent; (bottom) overall-consistent

N	D_W	D_B		
		2	4	8
64	0.3	0.051	0.053	0.066
128	0.4	0.05	0.076	0.079
256	0.9	0.1	0.1	0.2
512	1	0.05	0.23	0.23

N	D_O	f		
		2	4	8
64	0.3	308.70	4.87	4.77
128	0.4	265.74	4.38	4.38[a]
256	0.5	986.74	4.99[b]	4.26[a]
512	0.5	6557.1	1116.92[c]	591.84[c]

The ideal density in these cases, 0.89, is too high to distribute. The best density given these constraints is used instead, which leads to a worse performance of the standard reservoir

[a] $D_O = 0.3$

[b] $D_O = 0.5$

[c] $D_O = 0.1$

4.4 Results

4.4.1 NARMA-10

In this task, the optimal density D_O is consistent at 0.1 (Table 1).

In the overall consistency case, the results as summarised in the boxplots (Fig. 4a) show slightly better behaviour for the 4-subreservoir and 8-subreservoir ESNs 64 node case. There are no significant differences in results between the standard and 2–subreservoir ESNs of these sizes, however. For 128 and 256 nodes, the results are slightly worse for the rESNs, getting worse as the number of subreservoir increases. At 512 nodes, the results level out across all the ESN.

We hypothesise that the progression of these results, with rESNs working better in the smaller size, worse in the medium sizes, and equally well in the largest size may be explained by searching for an optimal reservoir structure

[4] The grid search is modified to split the range of f into 10 and use that as the initial step, and then split the range between the optimal value and its neighbour into 10 for the secondary step.

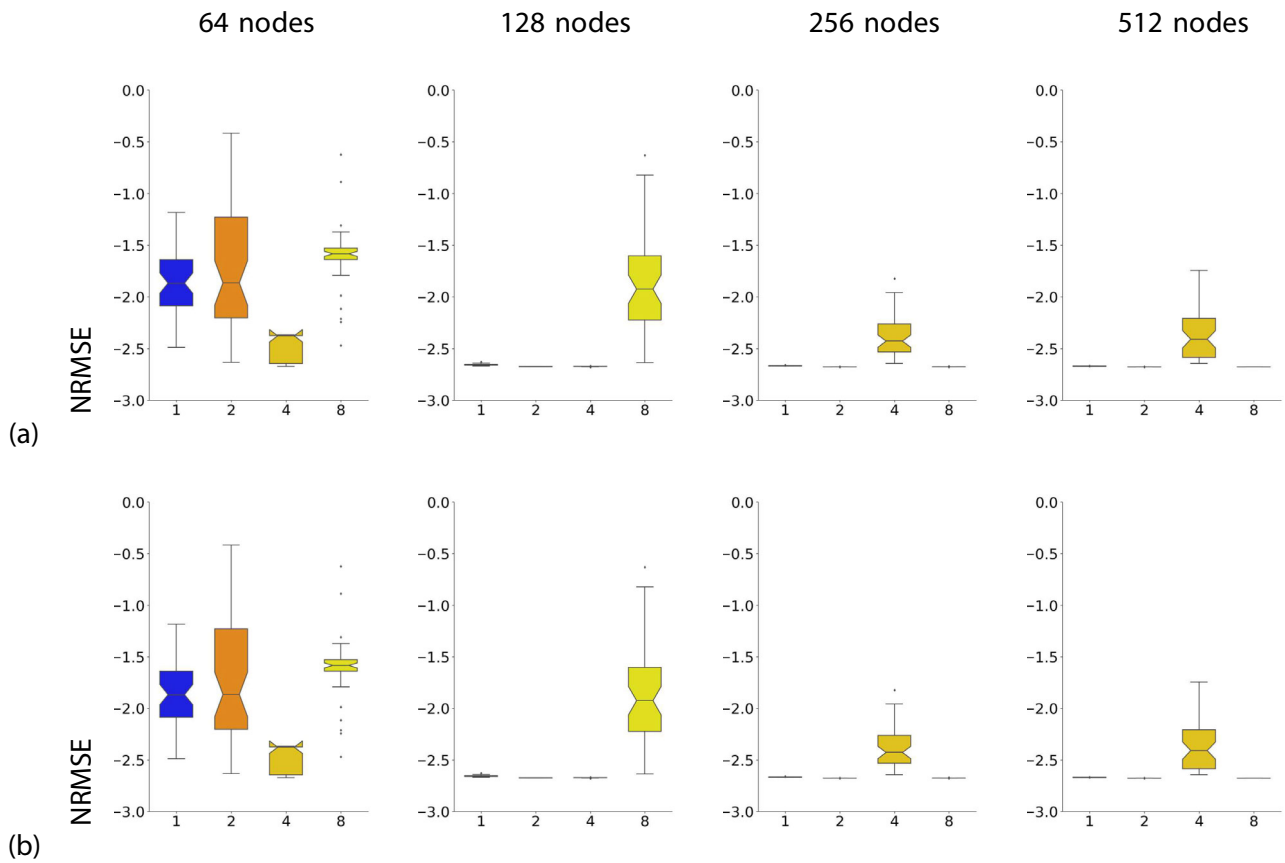

Fig. 4 The results for the NARMA-10 experiments, for row **a** overall consistency; row **b** patch consistency. In each chart, the *x* axis labels the number of subreservoirs (the standard reservoir is labelled '1'); the *y* axis is the NRMSE

using the *f* value. When searching for the optimal configuration of the restricted ESN, we find a maximal *f*-value, and then perform a two-level grid search between 1 and this maximum. The maximal *f*-value is smaller with smaller ESNs and with more subreservoirs, meaning that the search in these cases would be finer, and hence more likely to find a good result.

We hypothesise that there is therefore a greater chance of finding a good configuration in these smaller experiments. It may also follow that we could replicate these better results for larger ESNs by performing a more thorough search.

In the patch–consistent experiments we can observe that, for the 64 node case, the 4–subreservoir case leads to a worse performance, although the 2–subreservoir case is similar to the standard one. In the 128 and 256 node cases, we observe similar results across standard and restricted ESNs.

4.4.2 Sunspots

Unlike in the NARMA experiments, we observe no consistent optimal density across reservoir sizes; instead the optimal density increases with reservoir size (Table 2). We also observe that there is much less variation in performance across different ESN sizes (Fig. 5). It follows that any effect that restricting the ESN has will also, for the most part, be much smaller.

In the overall–consistent experiments, we observe little variation between the results from the standard and restricted ESNs, with the 4 and 8-subreservoir cases performing slightly better than the standard and 2-subreservoir one. However, as noted in Table 2, the ideal density in the 256-node case cannot be redistributed in an overall-consistent manner. Thus, while the restricted reservoirs in this case perform the same as their standard counterpart, this is not the optimal performance of a 256-node reservoir in this task. This effect gets worse in the 512-node task, as we cannot redistribute the connections between nodes so that the results do not diverge. As such, we do not report those results.

In the patch–consistent experiments, we observe similar results across configurations for all experiments.

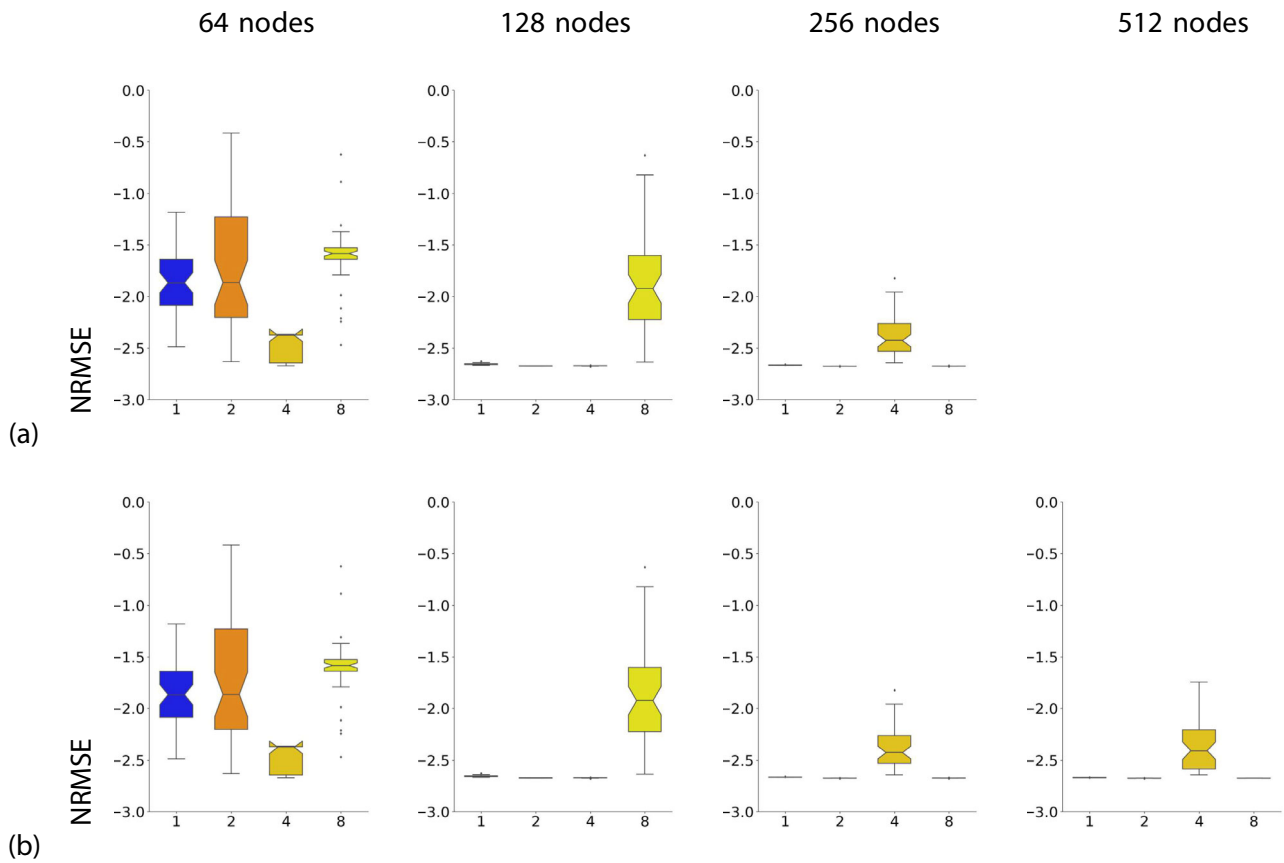

Fig. 5 The results for the Sunspots experiments, for row **a** overall consistency; row **b** patch consistency. In each chart, the *x* axis labels the number of subreservoirs (the standard reservoir is labelled '1'); the *y* axis is the NRMSE. The task requires a certain density to be possible; the highest achievable density for the 512–node case (see Table 2) lead to divergent results, so they have not been reported here

4.5 Conclusions

Throughout the experiments, there is no large difference between the standard and restricted ESNs. What few differences there are lessens as the ESNs grow larger, disappearing completely by the time we reach the 512-node case.

The more physically realistic of these models is the patch-consistent density. This model also has the advantage of not placing any constraints on the initial standard reservoir's density.

However, it is also the one with the more greater differences in performance in smaller sizes. This is particularly evident in the NARMA experiment, where the 8-subreservoir 64 node restricted ESN performs particularly badly.

When modelling these reservoirs, work may be needed to determine what makes a given subreservoir "reasonably large". We will therefore focus on these larger reservoirs in our future work.

Nevertheless, these results indicate that the restricted ESN model, using either overall or patch consistency, does not have a detrimental impact on performance when compared to a single large ESN. Hence restricted ESNs can form a suitable basis for building models of scaled-up reservoirs, heterogeneous reservoirs comprising subreservoirs of different materials, and for working on multiple timescale models.

5 MSO benchmark experiments

Having concluded that rESNs are a suitable basis to model larger reservoirs without a detrimental affect on performance, we now look at a more challenging task, which explicitly includes multiple timescales, in order to provide a baseline for future work on heterogeneous reservoirs. As such, we perform some experiments on two variations of the Multiple Superimposed Oscillators benchmark.

5.1 The MSO benchmark

The Multiple Superimposed Oscillators (MSO) task is a family of open system prediction benchmarks. The task

involves predicting the next value in a sequence generated by summing multiples of sines of the input. For MSO-n, the time series is defined by:

$$y(t) = \sum_{i=1}^{n} \sin(\alpha_i t) \tag{6}$$

where t is the timestep, and $\alpha = [0.2, 0.311, 0.42, 0.51, 0.63, 0.74, 0.85, 0.97]$. As n increases, more sine waves of increasing frequency are included in the sum; the specific α_i make these frequencies incommensurate, so the repetition time also increases with n. The $n = 2$ task was originally introduced in a presentation by Jaeger (Jaeger 2004) (what would now be called MSO-2, but there called "additive dynamics"). The task difficulty has been increased by extending the list of α with higher frequency values, to MSO-5 (Wierstra et al. 2005) and MSO-8 (Roeschies and Igel 2010).

5.2 Experimental setup

In order to provide a suitable baseline for future work, we modify the MSO benchmark as follows. The original benchmark is made harder by adding higher frequency components. Our long term aim is to investigate heterogeneous reservoirs with multiple timescales, with the fastest reservoir focussing on the highest frequency input component, and *lower* frequency sub-reservoirs focusing on lower frequency components, but without undersampling the higher frequency components.

Hence here we match the baseline frequency of the reservoir with the *maximum* frequency sine wave, given by $\alpha_8 = 0.97$. To do so, we sample Eq. (6) eight times more frequently (or, equivalently, reduce all the original MSO frequencies by a factor of eight):

$$y^*(t) = \sum_{i=1}^{n} \sin\left(\frac{\alpha_i t}{8}\right) \tag{7}$$

We further modify the task to scale the input to be between -0.5 and 0.5. We refer to this modified benchmark as MSO*. This is different from the original benchmark: the frequencies are lower, potentially making the task easier, but, given the same number of training and testing samples, less of the curve is sampled, potentially making the task harder. So we do not here compare results against other work; rather, we investigate the effect on performance of using (homogeneous) subreservoirs.

We use MSO*-2, 4 and 8. We base our dataset lengths of washout $= 100$, train $= 800$, test $= 200$ in the existing literature (Xue et al. 2007) and our preliminary experiments. These functions, along with the data lengths used, are shown in Fig. 6.

Continuing on from our conclusions in Sect. 4.5, we perform the multi–timescale experiments on patch–consistent rESNs. Instead of finding individual optimal densities as in the single–timescale experiments, we instead use $D_W = 0.005$ and $D_B = 0.001$, chosen using preliminary experiments using the methodology described in Sect. 4.

The experiments involve testing the MSO*-2, MSO*-4 and MSO*-8 tasks on a standard ESNs and rESNs with 2, 4, or 8 equal-sized subreservoirs, for ESNs of size $N \in [64, 128, 256, 512]$.

5.3 Results

As in our preliminary single-timescale experiments, we take the NRMSE of the output over 50 runs, which are shown in Fig. 7 (note that here we report the logarithm of NMSRE, as the results vary dramatically across systems). In all the experiments, we observe very similar behaviour in the standard ESN as it grows in size:

- best performance (lowest NRMSE) remains the same
- worst performance (highest NRMSE) improves

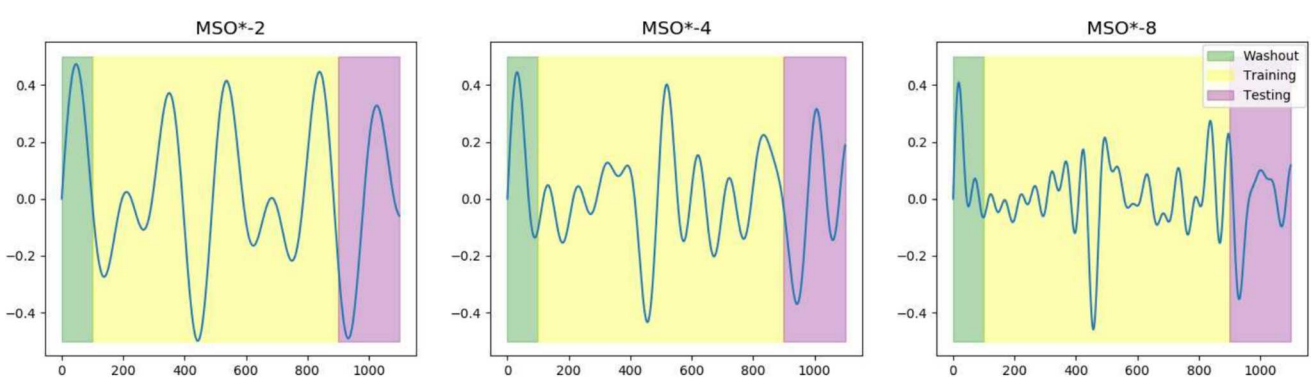

Fig. 6 Data used in our MSO* experiments (including range scaling), which each include 1100 datapoints (ranging from $t = 0$ to 138 in the original MSO equation). The coloured zones indicate the washout, training, and testing points

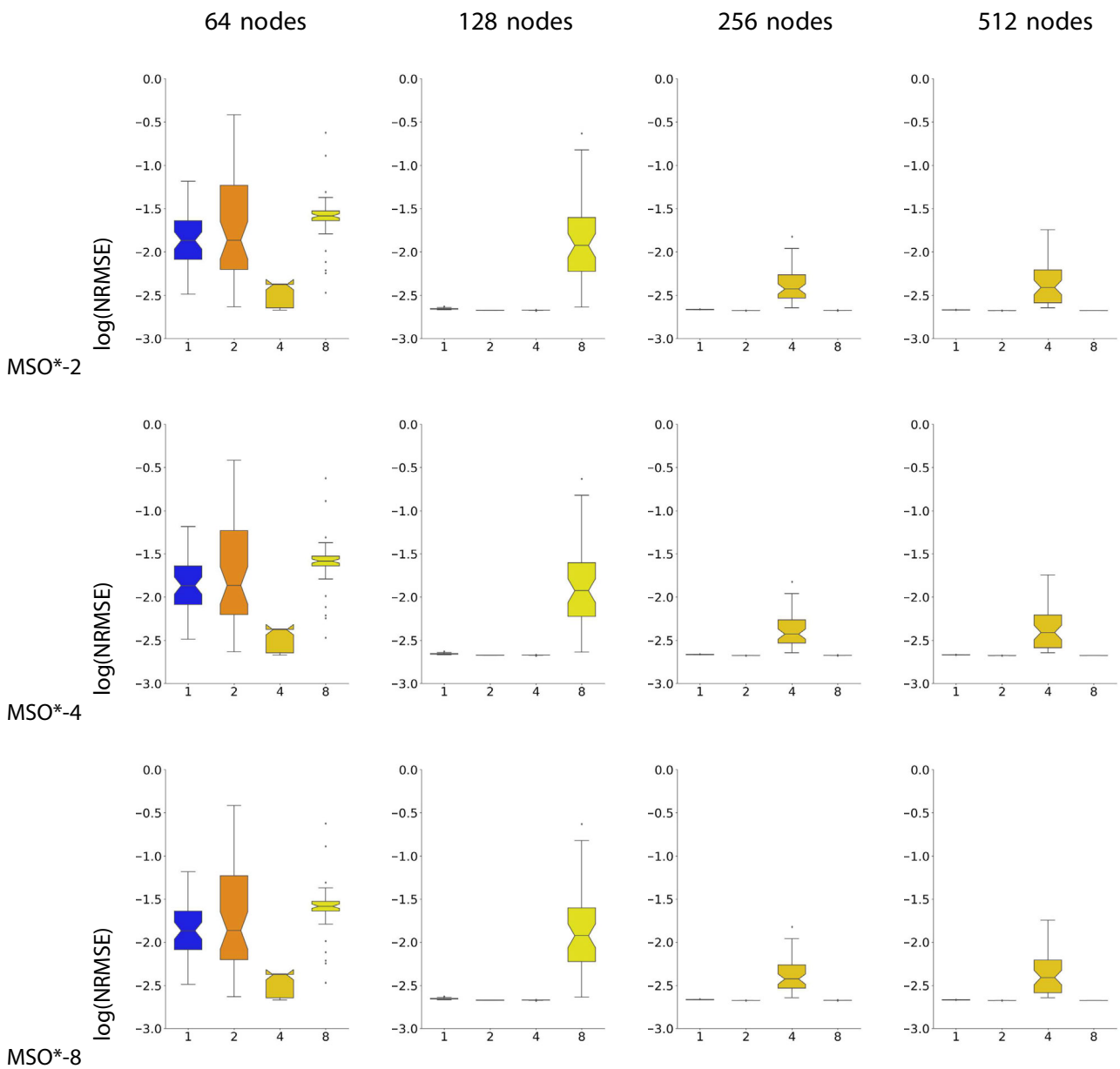

Fig. 7 The results for the MSO* experiments; the rows are the results for MSO*-2, MSO*-4, and MSO*-8. In each chart, the *x* axis labels the number of subreservoirs (the standard reservoir is labelled '1'); the *y* axis is $\log_{10}$(NRMSE)

We dub the best performance in this case the "saturation point", by analogy to the "saturation length" observed in certain benchmarks where increasing the training length has no effect on the performance of the reservoir (Dale 2018b).

The MSO*-2 experiments in Fig. 7 show that the performance of the rESNs is variable with the smaller reservoirs, but outperforms the standard reservoir as we increase the total number of nodes. In the 64-node case, only the 2-subreservoir restricted reservoir outperforms the standard reservoir, with the 4 and 8-subreservoir cases performing worse. By 512 nodes, however, all the reservoirs have reached the saturation point.

In the MSO*-8 experiments, we see a similar behaviour of the Standard ESN as we do in the MSO*-2 task, but with a saturation point which is higher than that of the MSO*-2 task. The MSO*-8 experiment also shows us a negative correlation in the 64-node case between the performance of the restricted reservoirs and the number of subreservoirs. This correlation also exists in the 128-node case, but not for the larger reservoirs. There, we see that all the restricted reservoirs outperform the standard one, and the number of subreservoirs has no effect on this performance.

No such patterns are apparent in the MSO*-4 experiments. There, we have very varied results in the 64-node case, followed by more consistent results in the larger reservoirs, with some exceptions. The 8-subreservoir 128 node restricted reservoir and the 4-subreservoir 256 and 512 node restricted reservoir have much worse results than the rest of the reservoirs. This inconsistency contrasts with the fairly regular results of the MSO*-2 and MSO*-8 experiments.

5.4 MSO* conclusions

Unlike the tasks in Sect. 4, we see some direct effects from restricting our ESNs. The effect does not appear to be consistent across the task size, leading to a better performance with MSO*-2 and MSO*-8 as we reach the larger subreservoir sizes, but have some odd outliers in the MSO*-4 case. We suggest that this effect may stem from the number of timescales involved in the task; in future work we will focus on decoupling subreservoirs by using different timescales.

6 Discussion and conclusions

Here we look at the effect that restricting larger reservoirs has on the performance of the reservoir on certain benchmark tasks, as a first step to determining whether joining smaller physical reservoirs together would be a good basis for scaling them up. We find that for more classical benchmark tasks like NARMA-10 and the Sunspots benchmark, there is very little effect that comes from restricting an ESN. With the more challenging task of MSO*, which explicitly incorporates multiple distinguishable timescales, though, this lack of effect no longer holds, and different complexity tasks respond to restriction in different ways.

Previous work (Xue et al. 2007) shows that a reservoir's performance at the MSO task can be improved upon using spatial decoupling within subreservoirs. We suggest that our inconsistent improvement or lack thereof comes from an analogous "accidental decoupling". In future work we will focus on ensuring the decoupling exists, focusing on temporal rather than spatial decoupling.

A Calculating D_W for the overall-consistent case

Given an ESN with N nodes and an average density $0 \leq D \leq 1$, we wish to restrict that ESN to have n subreservoirs of equal size; we assume n divides N. We set the density within the subreservoirs, D_W, to be greater than the density outside the subreservoirs by a factor of f, that is, $D_W = fD_B$.

In a restricted ESN with n subreservoirs, each of size N/n, there are n regions in the edge matrix $\mathbf{W}$ of size $(N/n)^2$ with density D_W, and a further $n^2 - n$ regions also of size $(N/n)^2$ with density D_B.

Hence the average density D of such a restricted ESN is:

$$D = \frac{nD_W + (n^2 - n)D_B}{n^2} \tag{8}$$

Substituting $D_W = fD_B$, and rearranging to get an expression for D_B in terms of D, we get:

$$D_B = \frac{Dn}{f + n - 1} \tag{9}$$

Once D_B is known, we also have D_W from $D_W = fD_B$.

B Optimising f

In order to find the best possible restricted ESN within our constraints, we optimise over the parameter f. However, we must somehow limit our search space.

In the restricted ESN, we want D_B to be strictly less than D_W (less dense connections than subreservoirs); therefore, $f > 1$.

To find an upper bound, we assume that every subreservoir is connected to every other subreservoir, that is, every connection weight matrix $\mathbf{B}_{ij}$ has at least one entry. This requires $D_B \geq (n/N)^2$. (In the experiments, the weight matrices are generated probabilistically, so when close to this density limit, it may be the case that there is not an edge between all subreservoirs.)

Rearranging Eq. 9 gives:

$$f = \frac{Dn}{D_B} - n + 1 \tag{10}$$

The lower limit on D_B gives an upper limit on f:

$$f \leq \frac{N^2 D}{n} - n + 1 \tag{11}$$

We also have an upper limit on the derived density, $D_W \leq 1$ (equality implies there are no zero elements in the relevant weight matrix). Substituting for D_W in Eq. 9 gives:

$$\frac{fDn}{f + n - 1} = D_W \leq 1 \tag{12}$$

Rearranging gives another upper limit on f:

$$f \leq \frac{n - 1}{Dn - 1} \tag{13}$$

Hence we have the upper and lower bounds on f:

$$1 < f \leq \min\left(\frac{N^2 D}{n} - n + 1, \frac{n-1}{Dn-1}\right) \tag{14}$$

Acknowledgements This work was made possible by PhD studentship funding from the Computer Science Department of the University of York.

Author contributions C.W. wrote the main manuscript text and performed all experiments. S.S. prepared Fig. 3 and contributed to Sect. 5.2. All authors developed the experiments and reviewed the manuscript.

Funding This work was made possible by PhD studentship funding from the Computer Science Department at the University of York

Data availibility statement The experimental results data can be found at https://github.com/FromAnkyra/modelling-restricting-resns-data/tree/main/results.The benchmark data for the NARMA and MSO* experiments was generated using the procedures described in the paper, and the code can be found at https://github.com/FromAnkyra/modelling-restricting-resns-data/tree/main/mso and https://github.com/FromAnkyra/modelling-restricting-resns-data/tree/main/narma.The benchmark data for the sunspots experiments can be found at https://machinelearningmastery.com/time-series-datasets-for-machine-learning/.

Declarations

Conflict of interest The authors have no Conflict of interest of a financial or personal nature, nor other interests that might be perceived to influence the results and/or discussion reported in this paper.

Ethical approval Not applicable, no human or animal research involved.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

Atiya AF, Parlos AG (2000) New results on recurrent network training: unifying the algorithms and accelerating convergence. IEEE TNN 11(3):697–709

Butcher JB, Verstraeten D, Schrauwen B, Haycock PW (2010) Extending reservoir computing with random static projections. ESANN 2010:303–308

Canaday D, Pomerance A, Gauthier DJ (2021) Model-free control of dynamical systems with deep reservoir computing. J Phys Complex 2(3):035025

Cucchi M, Abreu S, Ciccone G, Brunner D, Kleemann H (2022) Hands-on reservoir computing: a tutorial for practical implementation. Neuromorphic Comput Eng 2(3):032002

Dale M (2018a) Neuroevolution of hierarchical reservoir computers. In: GECCO 2018. ACM, pp 410–417

Dale M (2018b) Reservoir Computing in Materio. Ph.D. thesis, University of York

Dale M, O'Keefe S, Sebald A, Stepney S, Trefzer MA (2021) Computing with magnetic thin films: using film geometry to improve dynamics. In: UCNC 2021, volume 12984 of LNCS. Springer, pp 19–34

Deng Z, Zhang Y (2007) Collective behavior of a small-world recurrent neural system with scale-free distribution. IEEE TNN 18(5):1364–1375

Fernando C, Sojakka S (2003) Pattern recognition in a bucket. In: Advances in artificial life. Springer, pp 588–597

Gallicchio C, Micheli A (2011) Architectural and Markovian factors of echo state networks. Neural Netw 24(5):440–456

Gallicchio C, Micheli A (2017) Echo state property of deep reservoir computing networks. Cognit Comput 9(3):337–350

Gallicchio C, Micheli A, Pedrelli L (2017) Deep reservoir computing: a critical experimental analysis. Neurocomputing 268:87–99

Harding S, Miller J (2004) Evolution in materio: initial experiments with liquid crystal. In: Proceedings. 2004 NASA/DoD conference on evolvable hardware, pp 298–305

Iinuma T, Nobukawa S, Yamaguchi S (2022) Assembly of echo state networks driven by segregated low dimensional signals. In: 2022 international joint conference on neural networks (IJCNN), pp 1–8

Jaeger H (2001) The "echo state" approach to analysing and training recurrent neural networks - with an erratum note. In: German national research center for information technology GMD technical report, Bonn, Germany, vol 148(34), p 13

Jaeger H (2004) The echo state approach to recurrent neural networks (presentation). Accessed 29 Nov 2023

Jaeger H (2007) Discovering multiscale dynamical features with hierarchical echo state networks. Technical Report TR-10, Jacobs University Bremen

Jaeger H, Maass W, Principe J (2007) Special issue on echo state networks and liquid state machines. Neural Netw 20(3):287–289

Jarvis S, Rotter S, Egert U (2010) Extending stability through hierarchical clusters in echo state networks. Front Neuroinform, p 4

Kawai Y, Park J, Asada M (2023a) Reservoir computing using self-sustained oscillations in a locally connected neural network. Sci Rep 13(1):15532

Kawai Y, Park J, Tsuda I, Asada M (2023b) Learning long-term motor timing/patterns on an orthogonal basis in random neural networks. Neural Netw 163:298–311

Lukoševičius M (2012) A practical guide to applying echo state networks. In: LNCS. Springer, pp 659–686

Ma Q, Shen L, Cottrell GW (2017a) Deep-ESN: a multiple projection-encoding hierarchical reservoir computing framework. arXiv:1711.05255 [cs.LG]

Ma Q, Shen L, Zhuang W, Chen J (2017b) Decouple adversarial capacities with Dual-Reservoir network. In: ICONIP 2017. Springer, pp 475–483

Ma Q, Chen E, Lin Z, Yan J, Yu Z, Ng WWY (2021) Convolutional multitimescale echo state network. IEEE Trans Cybern 51(3):1613–1625

Maass W, Natschläger T, Markram H (2002) Real-time computing without stable states. Neural Comput 14(11):2531–2560

Malik ZK, Hussain A, Wu QJ (2017) Multilayered echo state machine: a novel architecture and algorithm. IEEE Trans Cybern 47(4):946–959

Rodan A, Tino P (2011) Minimum complexity echo state network. IEEE TNN 22(1):131–144

Rodriguez N, Izquierdo E, Ahn Y-Y (2019) Optimal modularity and memory capacity of neural reservoirs. Netw Neurosci 3(2):551–566

Roeschies B, Igel C (2010) Structure optimization of reservoir networks. Logic J IGPL 18(5):635–669

Schwenker F, Labib A (2009) Echo state networks and neural network ensembles to predict sunspots activity. In: ESANN 2009

Stepney S (2021) Non-instantaneous information transfer in physical reservoir computing. In: UCNC 2021 volume 12984 of LNCS. Springer, pp 164–176

Stepney S (2024) Physical reservoir computing: a tutorial. Nat Comput (submitted)

Tanaka G, Yamane T, Héroux JB, Nakane R, Kanazawa N, Takeda S, Numata H, Nakano D, Hirose A (2019) Recent advances in physical reservoir computing: a review. Neural Netw 115:100–123

Thompson A, Layzell P (1999) Analysis of unconventional evolved electronics. Commun ACM 42:71–79

Triefenbach F, Jalal A, Schrauwen B, Martens J-P (2010) Phoneme recognition with large hierarchical reservoirs. Adv Neural Inf Process Syst 23:2307–2315

Triefenbach F, Jalalvand A, Demuynck K, Martens J-P (2013) Acoustic modeling with hierarchical reservoirs. IEEE TASLP 21(11):2439–2450

Wierstra D, Gomez FJ, Schmidhuber J (2005) Modeling systems with internal state using evolino. In: GECCO 2025. ACM, pp 1795–1802

Wringe C, Stepney S, Trefzer MA (2023) Modelling and evaluating restricted ESNS. In: Genova D, Kari J (eds) UCNC 2023, volume 14003 of LNCS. Springer, pp 186–201

Xue Y, Yang L, Haykin S (2007) Decoupled echo state networks with lateral inhibition. Neural Netw 20(3):365–376

Yule GU (1927) On a method of investigating periodicities in disturbed series, with special reference to Wolfer's sunspot numbers. Philos Trans R Soc A 226(636–646):267–298

Zhang H, Vargas DV (2023) A survey on reservoir computing and its interdisciplinary applications beyond traditional machine learning. IEEE Access 11:81033–81070

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.